From Disjunctive Programs to Abduction*

Vladimir Lifschitz[1] and Hudson Turner[2]

[1] Department of Computer Sciences and Department of Philosophy
University of Texas at Austin
vl@cs.utexas.edu
[2] Department of Computer Sciences
University of Texas at Austin
hudson@cs.utexas.edu

Abstract. The purpose of this work is to clarify the relationship between three approaches to representing incomplete information in logic programming. Classical negation and epistemic disjunction are used in the first of these approaches, abductive logic programs with classical negation in the second, and a simpler form of abductive logic programming — without classical negation — in the third. In the literature, these ideas have been illustrated with examples related to properties of actions, and in this paper we consider an action domain also. We formalize this domain as a disjunctive program with classical negation, and then show how two abductive formalizations can be obtained from that program by a series of simple syntactic transformations. The three approaches under consideration turn out to be parts of a whole spectrum of different, but equivalent, ways of representing incomplete information.

1 Introduction

The purpose of this work is to clarify the relationship between three approaches to representing incomplete information in logic programming. Classical negation and epistemic disjunction are used in the first of these approaches ([GL91], [Tur94]), abductive logic programs with classical negation in the second [Gel91], and a simpler form of abductive logic programming — without classical negation — in the third [DDS93], [Dun93].

All these ideas have been illustrated with examples related to properties of actions, and in this paper we consider an action domain also. The theorems stated in this paper show, however, that the observations made here are of a general nature, and they may be applicable to other knowledge representation problems. Our example domain is the enhancement of Yale Shooting [HM87] in which two guns are available and it is assumed that at least one of them is loaded. This is a case of temporal projection with incomplete information about the initial situation.

After a brief review of the syntax and semantics of the relevant logic programming languages (Section 2), we formalize the two gun domain as a disjunctive

* This work was partially supported by the National Science Foundation under grant IRI-9306751. The second author is also supported by an IBM Graduate Fellowship.

program with classical negation, along the lines of [Tur94] (Section 3). Then, in Sections 4–9, we apply a series of six simple syntactic transformations to that program, so that a formulation in the spirit of [Gel91] is generated at one point along the way, and then a formulation in the spirit of [DDS93] is obtained. All programs formed in the process are equivalent to each other. Thus, the three approaches under consideration turn out to be parts of a whole spectrum of different, but equivalent, ways of representing incomplete information.

Each of Sections 4–9 concludes with a subsection called "Generalization," in which we present the underlying theorems, based for the most part on syntactic criteria, that are used to establish the correctness of the transformations. Thus, this sequence of theorems itself further illustrates the close relationships between the different approaches to representing incomplete information. Readers uninterested in the details of the underlying theorems can safely skip the "Generalization" subsections.

2 Rules, Constraints and Abduction

The answer set semantics [GL91] is defined for programs that consist of rules of the form

$$L_1 \mid \ldots \mid L_k \leftarrow L_{k+1}, \ldots, L_m, \mathit{not}\ L_{m+1}, \ldots, \mathit{not}\ L_n, \tag{1}$$

where $n \geq m \geq k \geq 0$, and each L_i is a literal (an atom possibly preceded by the classical negation sign $\neg$). If the "epistemic disjunction" $L_1 \mid \ldots \mid L_k$ in the head consists of a single literal ($k = 1$) then the rule is called *nondisjunctive*. If the head is empty ($k = 0$) then the rule is called a *constraint*. Given a rule r of the form (1), we define: $\mathit{head}(r) = \{L_1, \ldots, L_k\}$, $\mathit{pos}(r) = \{L_{k+1}, \ldots, L_m\}$, $\mathit{neg}(r) = \{L_{m+1}, \ldots, L_n\}$, and $\mathit{lit}(r) = \mathit{head}(r) \cup \mathit{pos}(r) \cup \mathit{neg}(r)$.

A program Π is *positive* if, for every rule $r \in \Pi$, $\mathit{neg}(r) = \emptyset$. The notion of an answer set is first defined for positive programs, as follows. A set X of literals is *closed under* a positive program Π if, for every rule $r \in \Pi$ such that $\mathit{pos}(r) \subset X$, $\mathit{head}(r) \cap X \neq \emptyset$.[1] A set of literals is *logically closed* if it is consistent or contains all literals. An *answer set* for a positive program Π is a minimal set of literals that is both closed under Π and logically closed.

Now let Π be an arbitrary program. Take a set X of literals. For each rule $r \in \Pi$ such that $\mathit{neg}(r) \cap X = \emptyset$, consider the rule r' defined by

$$\mathit{head}(r') = \mathit{head}(r),\ \mathit{pos}(r') = \mathit{pos}(r),\ \mathit{neg}(r') = \emptyset.$$

The positive program consisting of all rules r' obtained in this way is the *reduct* of Π relative to X, denoted by Π^X. We say that X is an *answer set* for Π if X is an answer set for Π^X.

It is clear from the definition of an answer set that adding a constraint to a program affects its meaning in a simple, "monotonic" way: this can only eliminate some of its answer sets. For instance, adding the constraint $\leftarrow \mathit{not}\ L$ to a program eliminates the answer sets that do not contain L.

[1] We write $X \subset Y$ when X is a subset of Y, not necessarily proper.

The definition of abduction accepted here follows Inoue and Sakama ([IS93], [IS94]), whose work is based on [KM90] and [Gel91]. An *abductive program* is a pair $\langle \Pi, \Gamma \rangle$, where Π is a program and Γ is a set of ground literals, called *abducibles*. A set X of ground literals is a *belief set* for an abductive program $\langle \Pi, \Gamma \rangle$ if it is a consistent answer set for the program $\Pi \cup (X \cap \Gamma)$.[2] The intersection $X \cap \Gamma$ is the *explanation* of X. When the set Γ of abducibles is clear from context, we will sometimes refer to an abductive program $\langle \Pi, \Gamma \rangle$ as Π.

3 A Disjunctive Program

There is a pilgrim and a turkey. The pilgrim has two guns, and at least one of them is initially loaded. The turkey is initially alive. The available actions are loading and shooting.

We are interested in representing this action domain using the situation calculus and negation as failure. Early work of this kind is described in [EK89], [Eva89], [AB90] and [Gel91]. The subject is treated more systematically in several recent papers, including [GL93], [DDS93] and [Dun93]. We assume that the reader is familiar with the main ideas of this work.

The following formulation of the two gun example is close to the one proposed in [Tur94]. It uses variables for situations (s), actions (a), propositional fluents (f) and guns (x).

We will call this program Π_0.

1. Initial conditions:

$$\begin{array}{l} Holds(Alive, S0) \leftarrow \\ Holds(Loaded(Gun1), S0) \mid Holds(Loaded(Gun2), S0) \leftarrow \end{array}$$

2. Effects of actions:

$$\begin{array}{l} Holds(Loaded(x), Result(Load(x), s)) \leftarrow \\ Noninertial(Loaded(x), Load(x), s) \leftarrow \\ \neg Holds(Alive, Result(Shoot(x), s)) \leftarrow Holds(Loaded(x), s) \\ Noninertial(Alive, Shoot(x), s) \leftarrow not\ \neg Holds(Loaded(x), s) \end{array}$$

3. Commonsense law of inertia:

$$\begin{array}{l} Holds(f, Result(a, s)) \leftarrow Holds(f, s), not\ \ Noninertial(f, a, s) \\ \neg Holds(f, Result(a, s)) \leftarrow \neg Holds(f, s), not\ \ Noninertial(f, a, s) \end{array}$$

4. Completeness rule:

$$Holds(f, S0) \mid \neg Holds(f, S0) \leftarrow$$

The last rule guarantees that each answer set X for Π_0 includes a complete description of the initial state of affairs: for every ground instance A of

[2] For notational convenience we sometimes identify a set Y of literals with the corresponding program $\{L \leftarrow\, : L \in Y\}$.

$Holds(f, S0)$, either A or $\neg A$ belongs to X. The answer sets for Π_0 are in a 1–1 correspondence with the possible initial states of the system.

Here is a somewhat more detailed description of the answer sets for Π_0. There are exactly 3 such sets: X_1, which includes the literals

$$Holds(Loaded(Gun1), S0) \text{ and } \neg Holds(Loaded(Gun2), S0),$$

X_2, which includes the literals

$$\neg Holds(Loaded(Gun1), S0) \text{ and } Holds(Loaded(Gun2), S0),$$

and X_3, which includes the literals

$$Holds(Loaded(Gun1), S0) \text{ and } Holds(Loaded(Gun2), S0).$$

Each answer set is complete, in the sense that, for every ground atom A that begins with *Holds*, either $A \in X_i$ or $\neg A \in X_i$.

4 Expressing Initial Conditions by Constraints

The first step in the sequence of syntactic transformations to be described here is the replacement of the first two rules of Π_0 — those that express the initial conditions — by the corresponding constraints:

$$\begin{array}{l} \leftarrow not\ Holds(Alive, S0), \\ \leftarrow not\ Holds(Loaded(Gun1), S0), not\ Holds(Loaded(Gun2), S0). \end{array}$$

We will denote the new program by Π_1.

Generally, the effect of a rule $A \leftarrow$ is quite different from the effect of the corresponding constraint $\leftarrow not\ A$. For instance, the only answer set for the program

$$\begin{array}{l} A \leftarrow \\ B \leftarrow A \end{array}$$

(where A and B are ground atoms) is $\{A, B\}$; whereas the program

$$\begin{array}{l} \leftarrow not\ A \\ B \leftarrow A \end{array}$$

has no answer sets. In the case of the transition from Π_0 to Π_1, however, the meaning of the program does not change:

Proposition 1. *Program Π_1 has the same answer sets as Π_0.*

This is not surprising. We have observed that the last rule of Π_0 forces each answer set to include a complete description of the initial state of the system. For this reason, the only effect of the initial conditions of Π_0 is to eliminate the answer sets corresponding to some of the initial states; these rules function as constraints.

Generalization

Given a rule r of form (1), let $constraint(r)$ denote the constraint

$$\leftarrow not\ L_1, \ldots, not\ L_k, L_{k+1}, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n\ . \tag{2}$$

Given a program Π, let

$$constraint(\Pi) = \{constraint(r) : r \in \Pi\}\ .$$

We'll consider a condition under which we can guarantee that a program $\Pi \cup \Pi'$ has the same consistent answer sets as the program $\Pi \cup constraint(\Pi')$.

For instance, take Π to be the program Π_0 minus the rules representing initial conditions, and take Π' to be $\Pi_0 \setminus \Pi$. So we have $\Pi_0 = \Pi \cup \Pi'$. Notice that we also have $\Pi_1 = \Pi \cup constraint(\Pi')$. What we wish to ensure is that the initial condition rules in Π_0 function exactly as the initial condition constraints in Π_1. We do this by showing that Π' and $constraint(\Pi')$ are interchangeable, in the sense that programs $\Pi \cup \Pi'$ and $\Pi \cup constraint(\Pi')$ have the same consistent answer sets.

We'll say that a set X of literals is *saturated* if every literal in X has its complement in X.

Given a set X of literals, we'll say that a set Y of literals is *complete in X* if for every literal $L \in X$ at least one of the complementary literals $L, \overline{L}$ belongs to Y.

Following is the definition from [Tur94] of a "signing" of a program.

Let Π be a constraint-free program, and let Lit denote the set of ground literals in the language of Π. Let S be a subset of Lit such that no literal in S that appears in the head of a rule in Π has its complement in the head of a rule in Π. We say that S is a *signing* for Π if each rule $r \in \Pi$ satisfies the following two conditions:

- $head(r) \cup pos(r) \subset S$ and $neg(r) \subset Lit \setminus S$, or
  $head(r) \cup pos(r) \subset Lit \setminus S$ and $neg(r) \subset S$,
- if $head(r) \subset S$, then $head(r)$ is a singleton.

Theorem 1. *Let Π and Π' be programs, and let Γ be a saturated set of literals such that $Lit \setminus \Gamma$ is a signing for the program $\Pi \cup \Pi'$. If every answer set for Π is complete in Γ and if the head of every rule in Π' is a subset of Γ, then programs $\Pi \cup \Pi'$ and $\Pi \cup constraint(\Pi')$ have the same consistent answer sets.*

We can use Theorem 1 to prove Proposition 1 as follows. Take Π and Π' as described above, so that $\Pi_0 = \Pi \cup \Pi'$ and $\Pi_1 = \Pi \cup constraint(\Pi')$. Let Γ be the set of all *Holds* literals. Clearly Γ is saturated. Furthermore, $Lit \setminus \Gamma$ is the set of all *Noninertial* literals, which is a signing for program Π_0. Every answer set for Π is complete in Γ, and for both rules $r \in \Pi'$ we have $head(r) \subset \Gamma$. So Theorem 1 guarantees that programs $\Pi \cup \Pi'$ and $\Pi \cup constraint(\Pi')$ have the same consistent answer sets.

Theorem 1 is proved using the following additional definitions, and theorem, from [Tur94].

Given rules r and r', we say that r *is subsumed by* r', and we write $r \preceq r'$, if the following three conditions hold:

- $neg(r') \subset neg(r)$,
- $pos(r') \subset pos(r)$,
- every literal in $head(r') \setminus head(r)$ appears complemented in $pos(r)$.

Given programs Π and Π', we say that Π *is subsumed by* Π', and we write $\Pi \preceq \Pi'$, if for each rule $r \in \Pi$ there is a rule $r' \in \Pi'$ such that $r \preceq r'$.

Given a program Π with signing S,

$$h_S(\Pi) = \{r \in \Pi : head(r) \subset S\} ,$$
$$h_{\overline{S}}(\Pi) = \{r \in \Pi : head(r) \subset Lit \setminus S\} .$$

Restricted Monotonicity Theorem. *Let Π, Π' be programs in the same language, both with signing S. If $h_{\overline{S}}(\Pi) \preceq h_{\overline{S}}(\Pi')$ and $h_S(\Pi') \preceq h_S(\Pi)$, then for every consistent answer set X' for program Π', there is a consistent answer set X for program Π such that $X \setminus S \subset X' \setminus S$.*

We'll also use the following lemma in the proof of Theorem 1.

Lemma 1. *Let Π be a program with signing S. A consistent set X of literals is an answer set for Π if and only if $X \setminus S$ is an answer set for program $h_{\overline{S}}(\Pi)^{X \cap S}$ and $X \cap S$ is the unique answer set for program $h_S(\Pi)^{X \setminus S}$.*

Proof. Let X be a consistent set of literals. Of course X is an answer set for Π if and only if X is an answer set for Π^X. Since S is a signing for Π, we have the following.

$$\begin{aligned} \Pi^X &= h_S(\Pi)^X \cup h_{\overline{S}}(\Pi)^X \\ &= h_S(\Pi)^{X \setminus S} \cup h_{\overline{S}}(\Pi)^{X \cap S} \end{aligned}$$

Observe that only literals from S appear in program $h_S(\Pi)^{X \setminus S}$, and similarly only literals not from S appear in program $h_{\overline{S}}(\Pi)^{X \cap S}$. It follows that X is an answer set for Π^X if and only if $X \setminus S$ is an answer set for program $h_{\overline{S}}(\Pi)^{X \cap S}$ and $X \cap S$ is an answer set for program $h_S(\Pi)^{X \setminus S}$. Finally, since program $h_S(\Pi)^{X \setminus S}$ is a nondisjunctive program without negation as failure, it has a unique answer set. □

Proof of Theorem 1. Assume that X is an answer set for $\Pi \cup constraint(\Pi')$. Thus, X is an answer set for Π that is closed under $constraint(\Pi')^X$. Since X is closed under $constraint(\Pi')^X$, it is also closed under $(\Pi')^X$. So X is closed under $\Pi^X \cup (\Pi')^X$. It follows that some subset X' of X is an answer set for $\Pi^X \cup (\Pi')^X$. So X' is a subset of X that is closed under Π^X. But X is an answer set for Π^X, so we can conclude that $X' = X$. Thus X is an answer set for $\Pi^X \cup (\Pi')^X = (\Pi \cup \Pi')^X$. That is, X is an answer set for program $\Pi \cup \Pi'$.

Assume that X' is a consistent answer set for program $\Pi \cup \Pi'$. Since X' is closed under $(\Pi')^X$, it is also closed under $constraint(\Pi')^X$. Notice that if

X' is also an answer set for Π, it will follow that X' is an answer set for $\Pi \cup constraint(\Pi')$. So we'll complete the proof by showing that X' is an answer set for Π. Let $S = Lit \setminus \Gamma$. Observe that $h_{\overline{S}}(\Pi) \preceq h_{\overline{S}}(\Pi \cup \Pi')$ and $h_S(\Pi \cup \Pi') \preceq h_S(\Pi)$. By the Restricted Monotonicity Theorem, there is a consistent answer set X for program Π such that $X \setminus S \subset X' \setminus S$. Thus, $X \cap \Gamma \subset X' \cap \Gamma$. But we know that X is complete in Γ and Γ is saturated, from which we can conclude that $X \cap \Gamma = X' \cap \Gamma$. It remains to show that $X \setminus \Gamma = X' \setminus \Gamma$. We can conclude by Lemma 1 that $X \setminus \Gamma$ is the unique answer set for program $h_S(\Pi)^{X \cap \Gamma}$. Similarly, $X' \setminus \Gamma$ is the unique answer set for program $h_S(\Pi \cup \Pi')^{X' \cap \Gamma}$. But $h_S(\Pi) = h_S(\Pi \cup \Pi')$, and $X \cap \Gamma = X' \cap \Gamma$; so we've shown that $X \setminus \Gamma = X' \setminus \Gamma$. □

5 Introducing Abduction

The next step is to turn Π_1 into an abductive program. The literals occurring in the ground instances of the completeness rule

$$Holds(f, S0) \mid \neg Holds(f, S0) \leftarrow \tag{3}$$

are declared abducibles, and the rule is replaced with the constraint

$$\leftarrow not\ Holds(f, S0), not\ \neg Holds(f, S0). \tag{4}$$

This constraint forces every explanation to contain one member of each complementary pair of abducibles.

The result is the following abductive program Π_2.

1. *Initial conditions:* (same as Π_1)

$$\leftarrow not\ Holds(Alive, S0)$$
$$\leftarrow not\ Holds(Loaded(Gun1), S0), not\ Holds(Loaded(Gun2), S0)$$

2. *Effects of actions:* (same as Π_0 and Π_1)

$$Holds(Loaded(x), Result(Load(x), s)) \leftarrow$$
$$Noninertial(Loaded(x), Load(x), s) \leftarrow$$
$$\neg Holds(Alive, Result(Shoot(x), s)) \leftarrow Holds(Loaded(x), s)$$
$$Noninertial(Alive, Shoot(x), s) \leftarrow not\ \neg Holds(Loaded(x), s)$$

3. *Commonsense law of inertia:* (same as Π_0 and Π_1)

$$Holds(f, Result(a, s)) \leftarrow Holds(f, s), not\ Noninertial(f, a, s)$$
$$\neg Holds(f, Result(a, s)) \leftarrow \neg Holds(f, s), not\ Noninertial(f, a, s)$$

4. *Completeness rule:*

$$\leftarrow not\ Holds(f, S0), not\ \neg Holds(f, S0)$$

Abducibles: The ground instances of $Holds(f, S0)$ and $\neg Holds(f, S0)$.

At this stage, epistemic disjunctions have been completely eliminated in favor of abduction: every rule of Π_2 is a nondisjunctive rule or a constraint.

The transition from Π_1 to Π_2 does not change the meaning of the program:

Proposition 2. *The belief sets for Π_2 are the same as the answer sets for Π_1.*

It is interesting to relate the process used in constructing Π_2 to the view of abduction developed by Inoue and Sakama [IS94]. They show that declaring a literal L abducible has the same effect as adding the rule

$$L \mid not\ L \leftarrow .$$

Such rules are syntactically different than the rules introduced in Section 2, because they contain the negation as failure operator in the head. However, the answer set semantics can be easily extended to rules like these [LW92]. In the case of Π_2, this approach to characterizing the abducibles leads to the rule

$$Holds(f, S0) \mid not\ Holds(f, S0) \leftarrow .$$

This rule differs from the completeness rule (3) only in that it uses negation as failure *not* instead of classical negation $\neg$. Proposition 2 shows that this difference can be offset by constraint (4).

The use of abduction in Π_2 is similar to the style of formalization proposed by Gelfond [Gel91]. One difference is that he does not introduce constraints like (4). (Without this constraint, Π_2 would have two additional belief sets, and the correspondence with the answer sets of Π_0 and Π_1 would be lost.)

Formulations of this kind are symmetric, in the sense that the rules expressing "positive facts" (such as the effect of *Load* on *Loaded*) look similar to the rules expressing "negative facts" (such as the effect of *Shoot* on *Alive*). This attractive feature will be lost in further transformations, but the final version will be in some ways more economical than the one presented in this section.

Generalization

In the presence of completeness of belief sets in a saturated set of abducibles, the interchangeability of abduction and disjunction is a general fact. Below we make this assertion precise.

Given a set X of literals, let

$$CC(X) = \{ \leftarrow not\ L, not\ \overline{L} : L \in X\} .$$

Notice that if we take X to be the set of abducibles for program Π_2 above, then the rules in $CC(X)$ are precisely the "completeness rules" (or completeness constraints) of Π_2.

If Π is an abductive program with set Γ of abducibles, adding $CC(\Gamma)$ to Π simply eliminates the beliefs sets that are not complete in Γ. That is, a set X of literals is a belief set for $\langle \Pi \cup CC(\Gamma), \Gamma \rangle$ if and only if X is a belief set for $\langle \Pi, \Gamma \rangle$ that is complete in Γ.

Given a set X of literals, let

$$DCR(X) = \{L \mid \overline{L} \leftarrow : L \in X\} .$$

Notice that if we take X to be the set of abducibles for program Π_2 above, then the rules in $DCR(X)$ are precisely the disjunctive "completeness rules" of program Π_1.

Theorem 2. *Let Π be a program, and Γ a saturated set of abducibles. The belief sets for the abductive program $\langle \Pi \cup CC(\Gamma), \Gamma \rangle$ are the same as the consistent answer sets for the program $\Pi \cup DCR(\Gamma)$.*

Proposition 2 is a special case of Theorem 2, in which we take Π to be the program Π_1 without the completeness rule.

Proof of Theorem 2 follows.

Lemma 2. *Let Π be a program, and Γ a saturated set of abducibles. A set X of literals is a belief set for $\langle \Pi \cup CC(\Gamma), \Gamma \rangle$ if and only if X is a consistent answer set for program $\Pi \cup (X \cap \Gamma)$ and $X \cap \Gamma$ is complete in Γ.*

Proof. We've already observed that X is a belief set for $\langle \Pi \cup CC(\Gamma), \Gamma \rangle$ if and only if X is a belief set for $\langle \Pi, \Gamma \rangle$ that is complete in Γ. Furthermore, by definition, X is a belief set for $\langle \Pi, \Gamma \rangle$ if and only if X is a consistent answer set for program $\Pi \cup (X \cap \Gamma)$. Finally, because Γ is saturated, we know that X is complete in Γ if and only if $X \cap \Gamma$ is complete in Γ. □

Lemma 3. *Let Π be a program, and Γ a saturated set of literals. A set X of literals is a consistent answer set for program $\Pi \cup DCR(\Gamma)$ if and only if X is a consistent answer set for program $\Pi \cup (X \cap \Gamma)$, with $X \cap \Gamma$ complete in Γ.*

Proof. Assume that X is a consistent answer set for program $\Pi \cup DCR(\Gamma)$. Thus X is an answer set for $\Pi^X \cup DCR(\Gamma)$. So X is closed under Π^X, and of course it's also closed under $X \cap \Gamma$. That is, X is closed under $\Pi^X \cup (X \cap \Gamma)$; and since X is consistent, some subset X' of X must be an answer set for $\Pi^X \cup (X \cap \Gamma)$. We know that X is closed under $DCR(\Gamma)$, from which it follows that X is complete in Γ. In fact, since Γ is saturated, we can conclude that $X \cap \Gamma$ is complete in Γ. It follows that any set closed under $\Pi^X \cup (X \cap \Gamma)$ must be complete in Γ. So X' is complete in Γ. But from this we can conclude that X' is also closed under $DCR(\Gamma)$. We already know that X' is closed under Π^X. Thus we see that X' is closed under $\Pi^X \cup DCR(\Gamma)$. It follows that $X' = X$, since X is an answer set for $\Pi^X \cup DCR(\Gamma)$. So X is an answer set for $\Pi^X \cup (X \cap \Gamma)$; and thus X is an answer set for $\Pi \cup (X \cap \Gamma)$.

Assume that X is a consistent answer set for program $\Pi \cup (X \cap \Gamma)$, with $X \cap \Gamma$ complete in Γ. So X is an answer set for $\Pi^X \cup (X \cap \Gamma)$, and thus X is closed under $\Pi^X \cup (X \cap \Gamma)$. It's clear that since X is complete in Γ, X is also closed under $DCR(\Gamma)$. We already know that X is closed under Π^X. Thus we see that X is closed under $\Pi^X \cup DCR(\Gamma)$. And since X is consistent, some subset X' of X is an answer set for $\Pi^X \cup DCR(\Gamma)$. But any set that is closed under $DCR(\Gamma)$ must be complete in Γ. Since X is consistent and complete in Γ, we can conclude that $X' \cap \Gamma = X \cap \Gamma$; so X' is closed under $X \cap \Gamma$. We know that X' is also closed under Π^X. Thus we see that X' is closed under $\Pi^X \cup (X \cap \Gamma)$. It follows that $X' = X$, since X is an answer set for $\Pi^X \cup (X \cap \Gamma)$. So X is an answer set for $\Pi^X \cup DCR(\Gamma)$; and thus X is an answer set for $\Pi \cup DCR(\Gamma)$. □

The theorem follows directly from the lemmas.

6 Eliminating Negative Abducibles

The explanation of every belief set for Π_2 contains exactly one element of each complementary pair of abducibles. In the next modification of the program, the instances of $\neg Holds(f, S0)$ are dropped from the set of abducibles, and constraint (4) is replaced with the "closed world assumption" (CWA) rule that generates these negative literals:

$$\neg Holds(f, S0) \leftarrow not\ Holds(f, S0).$$

Here is the resulting program Π_3.

1. *Initial conditions:* (same as Π_1 and Π_2)

$$\leftarrow not\ Holds(Alive, S0)$$
$$\leftarrow not\ Holds(Loaded(Gun1), S0), not\ Holds(Loaded(Gun2), S0)$$

2. *Effects of actions:* (same as $\Pi_0 - \Pi_2$)

$$Holds(Loaded(x), Result(Load(x), s)) \leftarrow$$
$$Noninertial(Loaded(x), Load(x), s) \leftarrow$$
$$\neg Holds(Alive, Result(Shoot(x), s)) \leftarrow Holds(Loaded(x), s)$$
$$Noninertial(Alive, Shoot(x), s) \leftarrow not\ \neg Holds(Loaded(x), s)$$

3. *Commonsense law of inertia:* (same as $\Pi_0 - \Pi_2$)

$$Holds(f, Result(a, s)) \leftarrow Holds(f, s), not\ Noninertial(f, a, s)$$
$$\neg Holds(f, Result(a, s)) \leftarrow \neg Holds(f, s), not\ Noninertial(f, a, s)$$

4. *Closed world assumption:*

$$\neg Holds(f, S0) \leftarrow not\ Holds(f, S0)$$

Abducibles: The ground instances of $Holds(f, S0)$.

Proposition 3. *Program Π_3 has the same belief sets as Π_2.*

Generalization

A saturated set Γ of abducibles along with the associated completeness constraints can always be replaced by a complete subset of Γ along with the appropriate closed world assumption rules. We make this precise below.

Given a set X of literals, let

$$CWA(X) = \{\overline{L} \leftarrow not\ L : L \in X\}$$

and let

$$\overline{X} = \{\overline{L} : L \in X\}\ .$$

Theorem 3. *Let Π be a program, and Γ a saturated set of abducibles. Let Γ' be a subset of Γ that is complete in Γ. The abductive program $\langle \Pi \cup CC(\Gamma), \Gamma \rangle$ has the same belief sets as the abductive program $\langle \Pi \cup CWA(\Gamma'), \Gamma' \rangle$.*

Proposition 3 is the special case of Theorem 3 in which we take Π to be the program Π_2 without the completeness rule, Γ to be the set of ground instances of $Holds(f, S0)$ and $\neg Holds(f, S0)$, and Γ' to be the set of ground instances of $Holds(f, S0)$.

Proof of Theorem 3 follows.

Lemma 4. *Let Π be a program, and Γ a saturated set of abducibles. Let Γ' be a complete subset of Γ. A set X of literals is a consistent answer set for $\Pi \cup CWA(\Gamma') \cup (X \cap \Gamma')$ if and only if X is a consistent answer set for program $\Pi \cup (X \cap \Gamma)$ and $X \cap \Gamma$ is complete in Γ.*

Proof. Assume that X is a consistent answer set for $\Pi \cup CWA(\Gamma') \cup (X \cap \Gamma')$. So X is closed under $\Pi^X \cup CWA(\Gamma')^X \cup (X \cap \Gamma')$. We need to show that X is an answer set for $\Pi \cup (X \cap \Gamma)$ and that $X \cap \Gamma$ is complete in Γ. Notice that $CWA(\Gamma')^X$ is $\overline{\Gamma' \setminus X}$; so $X \cap \overline{\Gamma'}$ is closed under $\overline{\Gamma' \setminus X}$. That is, $\overline{\Gamma' \setminus X} \subset X \cap \overline{\Gamma'}$. Since Γ' is complete in Γ, we have $\Gamma = \Gamma' \cup \overline{\Gamma'}$. So we see that $(X \cap \Gamma') \cup (\overline{\Gamma' \setminus X}) \subset X \cap \Gamma$. Furthermore it is easy to show, again using the completeness of Γ' in Γ, that $(X \cap \Gamma') \cup \overline{\Gamma' \setminus X}$ must be complete in Γ. It follows that $X \cap \Gamma$ is also complete in Γ. It remains to show that X is an answer set for $\Pi \cup (X \cap \Gamma)$. We already know that X is closed under Π^X. Thus X is closed under $\Pi^X \cup (X \cap \Gamma)$. Since X is consistent, some subset X' of X is an answer set for $\Pi^X \cup (X \cap \Gamma)$. Since X is consistent and $X \cap \Gamma$ is complete in Γ, and since X' is closed under $X \cap \Gamma$, we can can conclude that $X' \cap \Gamma = X \cap \Gamma$. It follows that X' is closed under $CWA(\Gamma')^X \cup (X \cap \Gamma')$. We already know that X' is closed under Π^X. Thus X' is closed under $\Pi^X \cup CWA(\Gamma')^X \cup (X \cap \Gamma')$. It follows that $X' = X$, since X is an answer set for $\Pi^X \cup CWA(\Gamma')^X \cup (X \cap \Gamma')$. So X is an answer set for $\Pi^X \cup (X \cap \Gamma)$; and thus X is an answer set for $\Pi \cup (X \cap \Gamma)$.

Assume that X is a consistent answer set for program $\Pi \cup (X \cap \Gamma)$, with $X \cap \Gamma$ complete in Γ. So X is closed under $\Pi^X \cup (X \cap \Gamma)$. Clearly X is closed under $X \cap \Gamma'$. As before, $CWA(\Gamma')^X$ is $\overline{\Gamma' \setminus X}$. Let L be a literal in $\overline{\Gamma' \setminus X}$. So $\overline{L} \in \Gamma' \setminus X$, and thus $\overline{L} \notin X$. Since $\Gamma' \subset \Gamma$, we have $\overline{L} \in \Gamma$. Since X is complete in Γ, we can conclude that L belongs to X. So we've shown that $\overline{\Gamma' \setminus X} \subset X$. That is, X is closed under $CWA(\Gamma')^X$. Since X is also closed under Π^X, we have shown that X is closed under $\Pi^X \cup CWA(\Gamma')^X \cup (X \cap \Gamma')$. Since X is consistent, some subset X' of X is an answer set for $\Pi^X \cup CWA(\Gamma')^X \cup (X \cap \Gamma')$. We can see that X' must contain $X \cap \Gamma'$. Since $CWA(\Gamma')^X$ is $\overline{\Gamma' \setminus X}$, we also see that X' must contain $\overline{\Gamma' \setminus X}$. Let L be a literal in $X \cap (\Gamma \setminus \Gamma')$. So $L \notin \Gamma'$, and since Γ' is complete in Γ, we see that $L \in \overline{\Gamma'}$. Thus $L \in \overline{\Gamma'} \cap X$. And since X is consistent, we can conclude that $L \in \overline{\Gamma' \setminus X}$. So X' also contains $X \cap (\Gamma \setminus \Gamma')$. Thus $X \cap \Gamma \subset X'$; so X' is closed under $X \cap \Gamma$. We already know that X' is closed under Π^X, so X' is closed under $\Pi \cup (X \cap \Gamma)$. It follows that $X' = X$, since X is an answer set for $\Pi \cup (X \cap \Gamma)$. So X is an answer set for $\Pi \cup CWA(\Gamma') \cup (X \cap \Gamma')$. □

Proof of Theorem 3. By Lemma 2 from the previous section, we know that X is a belief set for $\langle \Pi \cup CC(\Gamma), \Gamma \rangle$ if and only if X is a consistent answer set for program $\Pi \cup (X \cap \Gamma)$, with $X \cap \Gamma$ complete in Γ. By the definition of belief set, we know that X is a belief set for $\langle \Pi \cup CWA(\Gamma'), \Gamma' \rangle$ if and only if X is a consistent answer set for $\Pi \cup CWA(\Gamma') \cup (X \cap \Gamma')$. Now the theorem follows immediately from the lemma. □

7 Using the CWA to Generate Negative Facts

Every belief set for Π_3 contains exactly one element of each complementary pair of *Holds* literals. This "semantic" fact, along with several "syntactic" ones, allows us to replace two rules of Π_3 whose heads are negative with the closed world assumption rule

$$\neg Holds(f, s) \leftarrow not\ \ Holds(f, s).$$

This rule is more general than the closed world assumption of Π_3, so the latter can be dropped. Using the CWA, instead of specialized rules with negative heads, for generating negative conclusions seems to be the main distinctive feature of the use of abduction by Denecker and De Schreye [DDS93] and Dung [Dun93], in comparison with Gelfond's approach [Gel91].

The result of this transformation is the following program Π_4.

1. *Initial conditions:* (same as $\Pi_1 - \Pi_3$)

$$\leftarrow not\ \ Holds(Alive, S0)$$
$$\leftarrow not\ \ Holds(Loaded(Gun1), S0), not\ \ Holds(Loaded(Gun2), S0)$$

2. *Effects of actions:*

$$Holds(Loaded(x), Result(Load(x), s)) \leftarrow$$
$$Noninertial(Loaded(x), Load(x), s) \leftarrow$$
$$Noninertial(Alive, Shoot(x), s) \leftarrow not\ \ \neg Holds(Loaded(x), s)$$

3. *Commonsense law of inertia:*

$$Holds(f, Result(a, s)) \leftarrow Holds(f, s), not\ \ Noninertial(f, a, s)$$

4. *Closed world assumption:*

$$\neg Holds(f, s) \leftarrow not\ \ Holds(f, s)$$

Abducibles: The ground instances of $Holds(f, S0)$. (same as Π_3)

Proposition 4. *Program Π_4 has the same belief sets as Π_3.*

Generalization

A *level mapping* is a function from literals to ordinals. A program Π is *stratified* if it includes no constraints and if there is a level mapping f such that, for all rules $r \in \Pi$ and all literals L, L', the following three conditions are satisfied:

- if $L, L' \in head(r)$ then $f(L) = f(L')$,
- if $L \in head(r)$ and $L' \in pos(r)$ then $f(L) \geq f(L')$,
- if $L \in head(r)$ and $L' \in neg(r)$ then $f(L) > f(L')$.

We'll say that such a level mapping *stratifies* Π.[3]

Let Π be a nondisjunctive program. For ground literals L, L', we say that L *refers to* L' in Π if there is a rule in Π with L in the head and L' in the body. If the pair $\langle L, L' \rangle$ belongs to the reflexive transitive closure of the "refers to" relation, we say that L *depends on* L' in Π.[4]

We observe that if Π is a stratified nondisjunctive program, then a literal L depends on a literal L' in Π if and only if for every level mapping f that stratifies Π we have $f(L) \geq f(L')$.

Take Π to be the program Π_3 minus the rules for initial conditions. Let $R(L)$ denote the number of occurences of *Result* in a literal L. We define a level mapping f as follows:

$$f(L) = \begin{cases} 3R(L) & \text{, } \textit{if } L \textit{ is a ground instance of } Holds(f,s), \\ 3R(L)+1 & \text{, } \textit{if } L \textit{ is a ground instance of } \neg Holds(f,s), \\ 3R(L)+2 & \text{, } \textit{otherwise.} \end{cases}$$

It's easy to verify that f stratifies Π. Furthermore, we see that no *Holds* atom depends on its complement.

Theorem 4. *Let Π be a stratified nondisjunctive program, and let Γ be a set of abducibles such that for each consistent subset X of Γ, program $\Pi \cup X$ is consistent. Let C be a consistent set of literals such that every belief set for $\langle \Pi, \Gamma \rangle$ is complete in C. Let $\Pi' = \{ r \in \Pi : head(r) \not\subset C \}$. If no literal in $\overline{C}$ depends on its complement, then the abductive programs $\langle \Pi, \Gamma \rangle$ and $\langle \Pi' \cup CWA(\overline{C}), \Gamma \rangle$ have the same belief sets.*

Proposition 4 can be shown to follow from Theorem 4.

We omit the proof of Theorem 4, which relies on the fact that a stratified nondisjunctive program has at most one consistent answer set.

[3] The definition given here is equivalent to the usual definition of a "locally stratified" program [Prz88] when the set of atoms is defined as the set of ground atoms of a first-order language, and there is no classical negation.

[4] This definition extends in a natural way the standard notion of "depends on" for nondisjunctive programs without classical negation [ABW88]. In the following section, we further extend this notion to disjunctive programs (also with classical negation).

8 Eliminating Classical Negation in the Bodies of Rules

In the presence of the general closed world assumption included in Π_4, $\neg$*Holds* and *not Holds* are interchangeable in the bodies of rules, and *not* $\neg$ in front of *Holds* can be dropped in the bodies of rules.

Let Π_5 be the program obtained from Π_4 by dropping *not* $\neg$ from the last rule of Group 2.

Proposition 5. *Program Π_5 has the same belief sets as Π_4.*

We have eliminated classical negation from all rules other than the closed world assumption.

Generalization

Let C be a consistent set of literals. Given a rule r, let $r|C$ denote the rule obtained from r by replacing, for each literal $L \in C$, all occurences of *not* L in the body by $\overline{L}$ and all occurences of L in the body by *not* $\overline{L}$. Notice that no literal from C appears in the body of a rule $r|C$. More formally, $r|C$ is defined as follows:

- $head(r|C) = head(r)$,
- $pos(r|C) = (pos(r) \setminus C) \cup \overline{(neg(r) \cap C)}$,
- $neg(r|C) = (neg(r) \setminus C) \cup \overline{(pos(r) \cap C)}$.

We'll be interested in conditions guaranteeing that a program Π has the same consistent answer sets as the program $\{r|C : r \in \Pi\}$, for a consistent set C of literals. In order to state our theorem, we further extend the notion of "depends on" to programs with disjunction as well as classical negation, as follows.

A *splitting set* for a program Π is any set U of literals such that, for every rule $r \in \Pi$, if $head(r) \cap U \neq \emptyset$ then $lit(r) \subset U$.[5] For any program Π and literals L, L', we'll say that L *depends on* L' *in* Π if there is no splitting set U for Π with $L \in U$ and $L' \notin U$.

Theorem 5. *Let Π be a program with set Γ of abducibles. Let C be a consistent set of literals satisfying the following three conditions: no literal in C appears in Γ; no literal in $\overline{C}$ depends on its complement; and*

$$\{r \in \Pi : head(r) \cap C \neq \emptyset\} = CWA(\overline{C}) .$$

The abductive programs $\langle \Pi, \Gamma \rangle$ and $\langle \{r|C : r \in \Pi\}, \Gamma \rangle$ have the same belief sets.

Proposition 5 is a special case of Theorem 5. To see this, take Π to be Π_4, with Γ as in Π_4, and take C to be the set of ground instances of $\neg Holds(f, s)$.

The proof of Theorem 5 is based on a series of lemmas. The first of them allows us to replace literals $L \in C$ that occur in the body of a rule with *not* $\overline{L}$. Notice that in this lemma we needn't require that no literal in $\overline{C}$ depend upon its complement.

[5] This definition comes from [LT94].

Lemma 5. *Let Π be a program, and L a literal such that the only rule in Π with L in its head is the rule $L \leftarrow not\ \overline{L}$. For every rule $r \in \Pi$, let r_L be the rule such that $head(r_L) = head(r)$ and $pos(r_L) = pos(r) \setminus \{L\}$ and $neg(r_L) = neg(r) \cup \overline{(pos(r) \cap \{L\})}$. Let $\Pi_L = \{r_L : r \in \Pi\}$. Programs Π and Π_L have the same consistent answer sets.*

Proof. The proof proceeds in four parts.

(i) Assume that X is a consistent answer set for Π^Y. We'll show that X is closed under Π_L^Y. Notice that for all rules $r_L \in \Pi_L$, $r_L \neq r$ if and only if $L \in pos(r)$; so we need only consider rules $r_L \in \Pi_L$ such that $L \in pos(r)$. Let r_L be such a rule. Consider two cases. Case 1: $\overline{L} \in Y$. We know that $\overline{L}$ belongs to $neg(r_L)$, so $neg(r_L) \cap Y$ is nonempty and we're done. Case 2: $\overline{L} \notin Y$. Notice that L must belong to X, since Π includes the rule $L \leftarrow not\ \overline{L}$. If $neg(r_L) \cap Y$ is nonempty or $pos(r_L) \not\subset X$, we're done; so assume that $neg(r_L) \cap Y$ is empty and $pos(r_L) \subset X$. We must show that $head(r_L) \cap X$ is nonempty. To do this, we show that $neg(r) \cap Y$ is empty and $pos(r) \subset X$. Since $neg(r) \subset neg(r_L)$ and $neg(r_L) \cap Y$ is empty, we know that $neg(r) \cap Y$ is empty. We know that $pos(r) = pos(r_L) \cup \{L\}$, and that $pos(r_L) \subset X$. Since $L \in X$, we're done.

(ii) Assume that X is a consistent answer set for Π_L^Y. We'll show that X is closed under Π^Y. As before, for all rules $r \in \Pi$, $r \neq r_L$ if and only if $L \in pos(r)$; so we need only consider rules $r \in \Pi$ such that $L \in pos(r)$. Let r be such a rule. Consider two cases. Case 1: $\overline{L} \in Y$. Since the rule $L \leftarrow not\ \overline{L}$ is the only rule in Π_L with L in its head, we can conclude that L doesn't belong to X. Since $L \in pos(r)$, we have $pos(r) \not\subset X$; and we're done. Case 2: $\overline{L} \notin Y$. If $neg(r) \cap Y$ is nonempty or $pos(r) \not\subset X$, we're done; so assume that $neg(r) \cap Y$ is empty and $pos(r) \subset X$. We must show that $head(r) \cap X$ is nonempty. To do this, we show that $neg(r_L) \cap Y$ is empty and $pos(r_L) \subset X$. We know that $pos(r_L) \subset pos(r)$, so we conclude that $pos(r_L) \subset X$. We also know that $neg(r_L) = neg(r) \cup \{\overline{L}\}$. Since $\overline{L} \notin Y$ and $neg(r) \cap Y$ is empty, so is $neg(r_L) \cap Y$.

(iii) Assume that X is a consistent answer set for Π. By part (i), X is closed under Π_L^X; so some subset X' of X is an answer set for Π_L^X. By part (ii), X' is closed under Π^X. It follows that $X' = X$; so X is an answer set for Π_L.

(vi) Assume that X is a consistent answer set for Π_L. By part (ii), X is closed under Π^X; so some subset X of X is an answer set for Π^X. By part (i), X' is closed under Π_L^X. It follows that $X' = X$; so X is an answer set for Π. □

The next lemma uses the following additional definitions, and theorem, from [LT94].

Let U be a splitting set for a program Π. The set of rules $r \in \Pi$ such that $lit(r) \subset U$ is denoted by $b_U(\Pi)$. Let X be a set of literals. For each rule $r \in \Pi$ such that $pos(r) \cap U$ is a subset of X and $neg(r) \cap U$ is disjoint from X, take the rule r' defined by

$$head(r') = head(r),\ pos(r') = pos(r) \setminus U,\ neg(r') = neg(r) \setminus U.$$

The program consisting of all rules r' obtained in this way will be denoted by $e_U(\Pi, X)$. A *solution* to Π (with respect to U) is a pair $\langle X, Y\rangle$ of sets of literals such that

- X is an answer set for $b_U(\Pi)$,
- Y is an answer set for $e_U(\Pi \setminus b_U(\Pi), X)$,
- $X \cup Y$ is consistent.

Splitting Set Theorem. *Let U be a splitting set for a program Π. A set A of literals is a consistent answer set for Π if and only if $A = X \cup Y$ for some solution $\langle X, Y\rangle$ to Π with respect to U.*

Lemma 6. *Let Π be a program, and L a literal such that the only rule in Π with L in its head is the rule $L \leftarrow not\ \overline{L}$. Furthermore, assume that $\overline{L}$ does not depend on L in Π. For every rule $r \in \Pi$, let $r_{\overline{L}}$ be the rule such that $head(r_{\overline{L}}) = head(r)$ and $pos(r_{\overline{L}}) = pos(r) \cup \overline{(neg(r) \cap \{L\})}$, and $neg(r_{\overline{L}}) = neg(r) \setminus \{L\}$. Let $\Pi_{\overline{L}} = \{r_{\overline{L}} : r \in \Pi\}$. Programs Π and $\Pi_{\overline{L}}$ have the same consistent answer sets.*

Proof. Since $\overline{L}$ doesn't depend on L in Π, there is a splitting set U for Π such that $\overline{L}$ belongs to U while L does not. Notice that U is also a splitting set for program $\Pi_{\overline{L}}$, and that no rule in $b_U(\Pi)$ has L in its body. Thus we have $b_U(\Pi) = b_U(\Pi_{\overline{L}})$. What we will show is that for any consistent answer set X for program $b_U(\Pi)$, programs $e_U(\Pi \setminus b_U(\Pi), X)$ and $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)$ have the same consistent answer sets. Since $b_U(\Pi) = b_U(\Pi_{\overline{L}})$, it follows by the Splitting Set Theorem that programs Π and $\Pi_{\overline{L}}$ have the same consistent answer sets.

Assume that X is a consistent answer set for program $b_U(\Pi)$. The proof will proceed in four parts.

(i) Assume that Y is a consistent answer set for $e_U(\Pi \setminus b_U(\Pi), X)^Z$, with $Z \cap U$ empty, and with $L \in Z$ if and only if $L \in Y$. We'll show that Y is closed under $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)^Z$. What we must show is that for every rule $r_{\overline{L}} \in \Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}})$, if $pos(r_{\overline{L}}) \subset X \cup Y$ and $neg(r_{\overline{L}}) \cap (X \cup Z)$ is empty, then $head(r_{\overline{L}}) \cap Y$ is nonempty. Notice that for all rules $r_{\overline{L}} \in \Pi_{\overline{L}}$, $r_{\overline{L}} \neq r$ if and only if $L \in neg(r)$; so we need only consider rules $r_{\overline{L}} \in \Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}})$ such that $L \in neg(r)$. Let $r_{\overline{L}}$ be such a rule in $\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}})$. Consider two cases. Case 1: $L \in Y$. Notice that since L belongs to Y, we know that $\overline{L} \notin X$, because the rule $L \leftarrow not\ \overline{L}$ is the only rule in Π with L in its head. Since $L \in neg(r)$, we have $\overline{L} \in pos(r_{\overline{L}})$. Since $\overline{L} \notin X$ (and of course $\overline{L} \notin Y$), we have $pos(r_{\overline{L}}) \not\subset X \cup Y$ and we're done. Case 2: $L \notin Y$. Notice that since L doesn't belong to Y, we know that $\overline{L} \in X$, since the rule $L \leftarrow not\ \overline{L}$ occurs in Π. If $pos(r_{\overline{L}}) \not\subset X \cup Y$ or $neg(r_{\overline{L}}) \cap (X \cup Z)$ is nonempty, then we're done; so assume that $pos(r_{\overline{L}}) \subset X \cup Y$ and $neg(r_{\overline{L}}) \cap (X \cup Z)$ is empty. We need to show that $head(r_{\overline{L}}) \cap Y$ is nonempty. We do this by showing that $pos(r) \subset X \cup Y$ and $neg(r) \cap (X \cup Z)$ is empty. To begin, $pos(r) \subset pos(r_{\overline{L}})$, so we have $pos(r) \subset X \cup Y$. We also know that $neg(r) = neg(r_{\overline{L}}) \cup \{L\}$, so all that remains is to verify that L doesn't belong to $X \cup Z$. Since $L \notin Y$, we know that $L \notin Z$. And since $\overline{L} \in X$ and X is consistent, $L \notin X$. So $L \notin X \cup Z$.

(ii) Assume that Y is a consistent answer set for $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)^Z$, with $Z \cap U$ empty, and with $L \in Z$ if and only if $L \in Y$. We'll show that Y is closed under $e_U(\Pi \setminus b_U(\Pi), X)^Y$. What we must show is that for every rule $r \in \Pi \setminus b_U(\Pi)$, if $pos(r) \subset X \cup Y$ and $neg(r) \cap (X \cup Z)$ is empty, then $head(r) \cap Y$

is nonempty. Notice that for all rules $r \in \Pi$, $r \neq r_{\overline{L}}$ if and only if $L \in neg(r)$; so we need only consider rules $r \in \Pi \setminus b_U(\Pi)$ such that $L \in neg(r)$. Let r be such a rule. Consider two cases. Case 1: $L \in Y$. Since $L \in Y$, we know $L \in Z$. And since $L \in neg(r)$, we see that $neg(r) \cap (X \cup Z)$ is nonempty; so we're done. Case 2: $L \notin Y$. Notice that since $L \notin Y$, we know that $\overline{L} \in X$, since the rule $L \leftarrow not\ \overline{L}$ occurs in $\Pi_{\overline{L}}$. If $pos(r) \not\subset X \cup Y$ or $neg(r) \cap (X \cup Z)$ is nonempty, then we're done; so assume that $pos(r) \subset X \cup Y$ and $neg(r) \cap (X \cup Z)$ is empty. We need to show that $head(r) \cap Y$ is nonempty. To do this, we show that $pos(r_{\overline{L}}) \subset X \cup Y$ and $neg(r_{\overline{L}}) \cap (X \cup Z)$ is empty. To begin, $neg(r_{\overline{L}}) \subset neg(r)$, so we see that $neg(r_{\overline{L}}) \cap (X \cup Z)$ is empty. We know that $pos(r_{\overline{L}}) = pos(r) \cup \overline{(neg(r) \cap \{L\})}$, so all that remains is to verify that $\overline{L}$ belongs to $X \cup Y$, which is true since $\overline{L} \in X$.

(iii) Assume Y is a consistent answer set for program $e_U(\Pi \setminus b_U(\Pi), X)$. That is, Y is a consistent answer set for program $e_U(\Pi \setminus b_U(\Pi), X)^Y$. By part (i), Y is closed under program $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)^Y$, so some subset Y' of Y is an answer set for $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)^Y$. Notice that $L \in Y'$ if and only if $L \in Y$. So by part (ii), Y' is closed under program $e_U(\Pi \setminus b_U(\Pi), X)^Y$. It follows that $Y' = Y$; so Y is an answer set for $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)^Y$, which is to say that Y is an answer set for $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)$.

(iv) Assume Y is a consistent answer set for program $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)$. That is, Y is a consistent answer set for program $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)^Y$. By part (ii), Y is closed under program $e_U(\Pi \setminus b_U(\Pi), X)^Y$, so some subset Y' of Y is an answer set for $e_U(\Pi \setminus b_U(\Pi), X)^Y$. Notice that $L \in Y'$ if and only if $L \in Y$. So by part (i), Y' is closed under program $e_U(\Pi_{\overline{L}} \setminus b_U(\Pi_{\overline{L}}), X)^Y$. It follows that $Y' = Y$; so Y is an answer set for $e_U(\Pi \setminus b_U(\Pi), X)^Y$, which is to say that Y is an answer set for $e_U(\Pi \setminus b_U(\Pi), X)$. □

Lemma 7. *Let Π be a program, with C a consistent set of literals such that no literal in $\overline{C}$ depends on its complement and*

$$\{r \in \Pi : head(r) \cap C \neq \emptyset\} = CWA(\overline{C}) .$$

If L belongs to C, then programs Π and $\{r|\{L\} : r \in \Pi\}$ have the same consistent answer sets.

Proof. Follows by one application each of Lemmas 6 and 7. □

The following lemma is a consequence of Lemma 7.

Lemma 8. *Let Π be a program, with C a consistent set of literals such that no literal in $\overline{C}$ depends on its complement and*

$$\{r \in \Pi : head(r) \cap C \neq \emptyset\} = CWA(\overline{C}) .$$

Programs Π and $\{r|C : r \in \Pi\}$ have the same consistent answer sets.

Theorem 5 is an easy consequence of Lemma 8.

9 Dropping the CWA

Now we can eliminate classical negation altogether.

Let Π_6 be Π_5 without the closed world assumption rule:

1. Initial conditions: (same as $\Pi_1 - \Pi_5$)

$$\leftarrow not\ Holds(Alive, S0)$$
$$\leftarrow not\ Holds(Loaded(Gun1), S0), not\ Holds(Loaded(Gun2), S0)$$

2. Effects of actions: (same as Π_5)

$$Holds(Loaded(x), Result(Load(x), s)) \leftarrow$$
$$Noninertial(Loaded(x), Load(x), s) \leftarrow$$
$$Noninertial(Alive, Shoot(x), s) \leftarrow Holds(Loaded(x), s)$$

3. Commonsense law of inertia: (same as Π_4 and Π_5)

$$Holds(f, Result(a, s)) \leftarrow Holds(f, s), not\ Noninertial(f, a, s)$$

Abducibles: The ground instances of $Holds(f, S0)$. (same as $\Pi_3 - \Pi_5$)

This representation follows the method advocated in [DDS93].

The last transformation, unlike the ones performed earlier, does change the meaning of the program, of course. But the way the belief sets change when we drop the CWA is easy to describe.

In the following proposition, GA stands for the set of all ground atoms.

Proposition 6. *The map $X \mapsto X \cap GA$ establishes a 1–1 correspondence between the belief sets for Π_5 and the belief sets for Π_6. Moreover, each belief set X for Π_5 can be obtained from $X \cap GA$ by adding the negations of all ground atoms that begin with Holds and do not belong to $X \cap GA$.*

Generalization

The following theorem is an easy consequence of Proposition 2 from [LT94].

Theorem 6. *Let Π be a program and Γ a set of abducibles. Let C be a consistent set of literals such that no literal in C appears in Π or in Γ. If a set X of literals is a belief set for $\langle \Pi \cup CWA(\overline{C}), \Gamma \rangle$, then $X \setminus C$ is a belief set for $\langle \Pi, \Gamma \rangle$. Moreover, if X is a belief set for $\langle \Pi, \Gamma \rangle$, then $X \cup (C \setminus \overline{X})$ is a belief set for $\langle \Pi \cup CWA(\overline{C}), \Gamma \rangle$.*

Proposition 6 follows from Theorem 6. To see this, take Π to be Π_6, with Γ as in Π_6, and take C to be the set of ground instances of $\neg Holds(f, s)$.

10 Conclusion

We have presented a sequence of simple, syntactic, equivalence-preserving transformations of a program for reasoning about action. These transformations illustrate some of the close relationships between three approaches to representing incomplete information in logic programming: namely, disjunctive logic programs with classical negation; abductive logic programs with classical negation; and abductive logic programs without classical negation. The correctness of these transformations is proved by means of more general underlying theorems, which rely on various, mostly syntactic criteria that are not unique to programs for reasoning about action.

Among the seven programs in this paper formalizing the two gun domain, we prefer the program Π_2 presented in Section 5. Firstly, program Π_2 is symmetric, in the sense that the rules expressing "positive facts" (such as the effect of *Load* on *Loaded*) look similar to the rules expressing "negative facts" (such as the effect of *Shoot* on *Alive*). Secondly, on the basis of Lemma 2 (from Section 5), we can understand the abductive program Π_2 in terms of the family of simpler non-abductive programs $\Pi_2 \cup X$, where X is a set of literals representing a complete, consistent description of the initial situation.

Acknowledgments

The authors would like to thank Marc Denecker, Michael Gelfond and Norman McCain for useful discussions on the subject of this paper.

References

[AB90] Krzysztof Apt and Marc Bezem. Acyclic programs. In David Warren and Peter Szeredi, editors, *Logic Programming: Proc. of the Seventh Int'l Conf.*, pages 617–633, 1990.

[ABW88] Krzysztof Apt, Howard Blair, and Adrian Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 89–148. Morgan Kaufmann, San Mateo, CA, 1988.

[DDS93] Mark Denecker and Danny De Schreye. Representing incomplete knowledge in abductive logic programming. In Dale Miller, editor, *Logic Programming: Proceedings of the 1993 Int'l Symposium*, pages 147–163, 1993.

[Dun93] Phan Minh Dung. Representing actions in logic programming and its applications in database updates. In *Logic Programming: Proceedings of the Tenth Int'l Conf. on Logic Programming*, pages 222–238, 1993.

[EK89] Kave Eshghi and Robert Kowalski. Abduction compared with negation as failure. In Giorgio Levi and Maurizio Martelli, editors, *Logic Programming: Proc. of the Sixth Int'l Conf.*, pages 234–255, 1989.

[Eva89] Chris Evans. Negation-as-failure as an approach to the Hanks and McDermott problem. In *Proc. of the Second Int'l Symp. on Artificial Intelligence*, 1989.

[Gel91] Michael Gelfond. Epistemic approach to formalization of commonsense reasoning. Technical Report TR-91-2, University of Texas at El Paso, Department of Computer Science, 1991.

[GL91] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[GL93] Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *The Journal of Logic Programming*, 17:301–322, 1993.

[HM87] Steve Hanks and Drew McDermott. Nonmonotonic logic and temporal projection. *Artificial Intelligence*, 33(3):379–412, 1987.

[IS93] Katsumi Inoue and Chiaki Sakama. Transforming abductive logic programs to disjunctive programs. In *Logic Programming: Proceedings of the Tenth Int'l Conf. on Logic Programming*, pages 335–353, 1993.

[IS94] Katsumi Inoue and Chiaki Sakama. On positive occurrences of negation as failure. In *Proc. of the Fourth Int'l Conf. on Principles of Knowledge Representation and Reasoning*, pages 293–304, 1994.

[KM90] Antonis Kakas and Paolo Mancarella. Generalized stable models: a semantics for abduction. In *Proc. of ECAI-90*, pages 385–391, 1990.

[LT94] Vladimir Lifschitz and Hudson Turner. Splitting a logic program. In *Logic Programming: Proceedings of the Eleventh Int'l Conf. on Logic Programming*, pages 23–37, 1994.

[LW92] Vladimir Lifschitz and Thomas Woo. Answer sets in general nonmonotonic reasoning (preliminary report). In Bernhard Nebel, Charles Rich, and William Swartout, editors, *Proc. of the Third Int'l Conf. on Principles of Knowledge Representation and Reasoning*, pages 603–614, 1992.

[Prz88] Teodor Przymusinski. On the declarative semantics of deductive databases and logic programs. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 193–216. Morgan Kaufmann, San Mateo, CA, 1988.

[Tur94] Hudson Turner. Signed logic programs. In *Logic Programming: Proceedings of the 1994 Int'l Symposium*, pages 61–75, 1994.