Reasoning in Open Domains

Michael Gelfond
Computer Science Department
University of Texas at El Paso
El Paso, Texas 79968
mgelfond@cs.ep.utexas.edu

Halina Przymusinska
Computer Science Department
California State University
Pomona, Ca. 91768
halina@ucrengr.ucr.edu

Abstract

In this paper we modify the semantics of epistemic specifications (and hence the answer set semantics of extended logic program and disjunctive databases) to allow for **reasoning in the absence of domain–closure assumption.** This modification increases the expressive power of the language and allows one to explicitly state the domain–closure and other assumptions about the domain of discourse in the language of epistemic specifications. The power of the language is demonstrated by way of examples. In particular we show how open domain assumption can be used to formalize default reasoning in the presence of anonymous exceptions to defaults.

1 Introduction

Epistemic specifications were introduced in [4] as a tool for knowledge representation. They can be viewed as a generalization of "extended disjunctive databases" from [6] capable of expressing powerful forms of introspection. The semantics of an epistemic specification Π has been given via the notion of a *world view* of Π - a collection of vivid theories about the world [8] which can be built by a rational reasoner on the instructions from Π. The concept of a world view of Π was defined in two steps: first the rules from Π were replaced by their ground instances, and then the definition of a world view was given for specifications not containing variables. Equating a specification Π with the set of its ground instances which occurs during the first step was justified by the domain closure assumption [13] which asserts that all objects in the domain of discourse have names in the language of Π. Even though the assumption is undoubtedly useful for a broad range of applications there are cases when it does not properly reflect the properties of the domain of discourse. In this paper we modify the semantics of epistemic specifications from [4] (and hence the answer set semantics of extended logic program and disjunctive databases) to allow for **reasoning**

in the absence of domain–closure assumption. This modification increases the expressive power of the language and allows one to explicitly state the domain–closure and other assumptions about the domain of discourse in the language of epistemic specifications. [1] The paper is organized as follows: section two contains the definitions of syntax and semantics of epistemic specifications, and section three contains various formalizations of domain–closure, unique names, and closed world assumptions. As another example of applicability of the new semantics to knowledge representation we discuss a possible solution to the problem of anonymous extensions to defaults [2]. Readers who wish to ignore global introspection and restrict their reading to extended logic programs can do so by examining the first two steps of the definition in section two and replacing all occurrences of $\neg Mp$ in section three by *not* p.

2 Definitions

Let us consider a language $\mathcal{L}$ consisting of predicate symbols $p, q, \ldots$, object variables, function symbols, a boolean constant **true**, connectives &, $\neg$, $\exists$, and the modal operators K and M, (where KF stands for "F is known to be true," and MF stands for "F may be believed to be true"). Terms and formulae of $\mathcal{L}$ will be defined in the usual way. Formulae of the form $p(t_1, \ldots, t_n)$ where t_i is a term are called atoms. (Parentheses will be skipped in some cases for convenience.) By literals we will mean atoms $p(t_1, \ldots, t_n)$ and their negations $\neg p(t_1, \ldots, t_n)$. Formulae not containing free variables are called ground formulae (or statements). The set of all ground literals will be denoted by Lit. Let us consider a collection $A = \{A_i\}$ of sets of ground literals and a set W of such literals. (A can be thought of as a collection of possible belief sets of a reasoner while W represents his current (working) set of beliefs.) We will inductively define the notion of truth ($\models$) and falsity ($=\!|$) of formulae of $\mathcal{L}$ w.r.t. a pair $M = < A, W >$.

$M \models p(t_1, \ldots, t_n)$ iff $p(t_1, \ldots, t_n) \in W$ where $t_1, \ldots, t_n$ are ground terms from $\mathcal{L}$

$M \models \mathrm{K}F$ iff $< A, A_k > \models F$ for every A_k from A

$M \models \mathrm{M}F$ iff $< A, A_k > \models F$ for some A_k from A

$M \models F\&G$ iff $M \models F$ and $M \models G$

$M \models \exists x F$ iff there is a ground term t from $\mathcal{L}$ such that $M \models F(t)$

$M \models \neg F$ iff $M =\!| F$

[1] This can be viewed as a further development of the program started in [5], [4] where the language of general logic programs has been expended to explicitly express the closed world assumption.

$M =| p(t_1, \ldots, t_n)$ iff $\neg p(t_1, \ldots, t_n) \in W$

$M =| \mathrm{K}F$ iff $M \not\models \mathrm{K}F$

$M =| \mathrm{M}F$ iff $M \not\models \mathrm{M}F$

$M =| F \& G$ iff $M =| F$ or $M =| G$

$M =| \exists x F$ iff for every ground term t from $\mathcal{L}$, $M =| F(t)$

$M =| \neg F$ iff $M \models F$

In our further discussion we will expand language $\mathcal{L}$ by the connectives *or* and $\forall$ defined as follows:

$(F \; or \; G)$ iff $\neg(\neg F \,\&\, \neg G)$ $\qquad$ $\forall x F$ iff $\neg \exists x \neg F$

Formulae of the expanded language $\mathcal{L}_0$ not containing modal operators will be called *objective formulae*. Formulae constructed from $\mathrm{K}F$ and $\mathrm{M}F$ (where F is objective) and from logical connectives and quantifiers will be called *subjective*. It is easy to see that according to the definition above the truth of subjective sentences does not depend on W while the truth of objective ones does not depend on A, i.e. we have a notion of objective formula being true (false) in W and subjective formula being true (false) in A. We will denote the former by $W \models F$ ($W =| F$) and later by $A \models F$ ($A =| F$). Notice also, that the definition of truth depends significantly on the alphabet of $\mathcal{L}$. A statement $\forall x Px$ is true in the set $W = \{Pa\}$ if the alphabet of the corresponding language is $\{a\}$ and false otherwise.

By an *epistemic specification* we will mean a collection of rules of the form

$$F \leftarrow G_1, \ldots, G_m, \mathit{not}\; G_{m+1}, \ldots, \mathit{not}\; G_k \tag{1}$$

where F and $G_{m+1} \ldots G_k$ are objective and $G_1 \ldots G_m$ are subjective or objective formulae. [2]

Now let Π be an epistemic specification over a language $\mathcal{L}_0$. To give a semantics of Π we will first expand the alphabet of $\mathcal{L}_0$ by an infinite sequence of new constants $c_1, \ldots, c_k, \ldots$. We will call these new constants *generic*. The resulting language will be denoted by $\mathcal{L}_\infty$. By $\mathcal{L}_k$ we will denote the expansion of $\mathcal{L}_0$ by constants $c_1, \ldots, c_k$. Π_k, where $0 \leq k \leq \infty$, will stand for the set of all ground instances of Π in the language $\mathcal{L}_k$. The truth relation in the language $\mathcal{L}_k$ will be denoted by $\models_k$. The index will be omitted whenever possible. Now we will define the notion of a *k–belief set* for any nonnegative integer k and any epistemic specification Π. The definition will follow in several steps.

Step 1. Let us first assume that Π is an epistemic specification **not containing modal operators and negation as failure**. A pair $< k, B >$

[2] We will assume that every epistemic specification contains a rule **true** $\leftarrow$.

where k is a nonnegative integer and B is a set of ground literals in $\mathcal{L}_k$ will be called a *k–belief set* of Π iff B is a minimal set satisfying the following two conditions:

1. For every rule $F \leftarrow G_1, \ldots, G_m$ from Π_k such that $B \models_k G_1 \,\&\, \ldots \,\&\, G_m$ we have $B \models_k F$.

2. if B contains a pair of complementary literals then $B = Lit$.

Example 1. Consider a specification $\Pi^0 = \{Pa \leftarrow\}$. Assume that the alphabet of the language $\mathcal{L}_0$ of Π^0 is $\{a\}$. For any k the k–belief set of Π^0 is $< k, \{Pa\} >$. The k–belief sets of the specification Π^1 obtained by adding to Π^0 a rule $\exists x Qx \leftarrow$ have the form $< k, \{Pa, Qa\} >, \ldots < k, \{Pa, Qc_k\} >$.

Step 2. Now let us assume that Π is an epistemic specification **not containing modal operators** and let B be a set of ground literals in the language $\mathcal{L}_k$. By Π_k^B we will denote the result of

1. removing from Π_k all the rules containing formulae of the form *not* G such that $B \models_k G$

2. removing from the rules in Π_k all other occurrences of formulae of the form *not* G.

Obviously, Π_k^B contains neither modal operators nor negation by failure and therefore its belief sets are defined in step one. We will say that $< k, B >$ is a **k–belief set** of Π if $< k, B >$ is a k–belief set of Π_k^B.

Example 2. Let us view rules $Pa \leftarrow, \quad Qa \leftarrow$ *not* Px as an epistemic specification Π over the language $\mathcal{L}_0$ with the alphabet $\{a\}$. It is easy to see that k–belief set of Π is $< k, \{Pa\} >$ if $k = 0$ and $< k, \{Pa, Qa\} >$ otherwise.

Step 3. Now we are ready to define a world view of an **arbitrary** epistemic specification Π over a language $\mathcal{L}$. Let $\mathbf{A_k}$ be a collection of sets of ground literals in the language $\mathcal{L}_k$ and let $\Delta_k = \{< k, B >: \; B \in \mathbf{A_k}\}$. By $\Pi^{\mathbf{A_k}}$ we will denote the epistemic specification obtained from Π_k by:

1. removing from Π_k all rules containing formulae of the form G such that G is subjective and $\mathbf{A_k} \not\models_k G$.

2. removing from the rules in Π_k all other occurrences of subjective formulae.

We will say that Δ_k is a **k-world view** of Π if

1. Δ_k is not empty,

2. Δ_k is the set of all consistent k-belief sets [3] of $\Pi^{\mathbf{A_k}}$.

[3] $< k, B >$ is consistent if B does not contain contrary literals.

Elements of Δ_k will be called **k-belief sets** of Π. (Whenever the indeces are irrelevant to the discussion we will omit them.)

Example 3. Consider again the rules from Example 2 and expand them by two more rules

3. $\neg Px \leftarrow \neg \mathrm{M}Px$ 4. $\neg Qx \leftarrow \neg \mathrm{M}Qx$

expressing the closed world assumptions for predicates P and Q [4].

It is easy to see that the following are the world views of Π.

$\{< 0, \{Pa, \neg Qa\} >\}$,

$\{< 1, \{Pa, Qa, \neg Pc_1, \neg Qc_1, \} >\}$,

$\{< 2, \{Pa, Qa, \neg Pc_1, \neg Qc_1, \neg Pc_2, \neg Qc_2, \} >\}$,

...

We will say that an epistemic specification Π is **consistent** if it has a world view. Π *entails* a statement F of $\mathcal{L}_0$ ($\Pi \models F$) if F is true in all world views of Π. Π answers *yes* to a query Q if $\Pi \models Q$, *no* if $\Pi \models \neg Q$, and *unknown* otherwise. In the example above Π answers *yes* to a query Pa, *no* to a query $\neg Pa$ and *unknown* to a query Qa.

Example 4. Consider a language $\mathcal{L}_0$ over the alphabet $\{a\}$ and a specification Π from Example 1, consisting of the rule $Pa \leftarrow$.

The following are world views of Π:
$\{< 0, \{Pa\} >\}$,$\{< 1, \{Pa\} >\}$, $\{< 2, \{Pa\} >\}$, ...

Obviously, as intended, Π's answer to a query $\forall x Px$ is *unknown*.

3 Applications

3.1 Domain Assumptions

In the previous section we removed assumptions about the domain of discourse from the semantics of epistemic specifications. Now we will demonstrate how these assumptions can be expressed in our language. Let Π be an arbitrary epistemic specification in a language $\mathcal{L}_0$. We expand $\mathcal{L}_0$ by the unary predicate symbol H which stand for *named elements of a domain*. The following rules can be viewed as the definition of H:

H_1. $Ht \leftarrow$ (for every ground term t from $\mathcal{L}_0$)

H_2. $\neg Hx \leftarrow not\ Hx$

The domain–closure assumption can be expressed by the rule:

DCA. $\leftarrow \exists x \neg Hx$ [4]

[4] A rule $\leftarrow \Gamma$ with empty conclusion is a shorthand for the rule $\neg true \leftarrow \Gamma$. Rules of this sort prohibit the reasoner from believing in Γ and differ from $\neg\Gamma \leftarrow$ which assert that Γ is false.

The extension of Π by the rule DCA will be denoted by Π_C. The following propositions may help one to better understand this rule:

Proposition 1. For any specification Π not containing predicate H and any query Q, $\Pi_C \models Q$ iff $\Pi_0 \models_0 Q$.

Example 5. Let Π be a specification from Example 3 expanded by the rules H_1 and H_2. The k-world view of Π looks as follows:

$\{< 0, \{Ha, Pa, \neg Qa\} >\}$, if $k = 0$

$\{< k, \{Ha, \neg Hc_1 \ldots \neg Hc_k, Pa, Qa, \neg Pc_1, \neg Qc_1 \ldots \neg Pc_k, \neg Qc_k, \} >\}$, if $k > 0$

and therefore Π's answer to a query Qa is *unknown*. The answer changes if Π is expanded by the domain–closure assumption. Π_C has the unique world view $\{< 0, \{Ha, Pa, \neg Qa\} >\}$ and therefore the Π_C's answer to Qa is *no*, exactly the answer produced by stable model semantics. This observation is generalized by the following proposition:

Proposition 2. Let Π be a general logic program with unique stable model and let Π_t be the epistemic specification obtained from Π by adding to it the closed world assumption $\neg Px \leftarrow \neg \mathrm{M}Px$ and the domain–closure assumption DCA. Then for every query Q, Π and Π_t produce the same answers to Q.

Now we will briefly discuss several examples of the use of domain assumptions and of the concept of known objects to formalization of commonsense reasoning.

Example 6. Consider the departmental database containing the list of courses which will be offered by the department next year and the list of professors who will be working for the department at that time. Let us assume that the database knows the names of all the courses which may be taught by the department but, since the hiring process is not yet over, it does not know the names of all of the professors. This information can be expressed as follows:

$course(a) \leftarrow \qquad course(b) \leftarrow$

$prof(m) \leftarrow \qquad prof(n) \leftarrow$

$\neg\ course(x) \leftarrow \neg Hx$

The k-world view of this specification is

$< k, \{course(a), course(b), \neg\ course(c_1) \ldots \neg course(c_k), prof(m), prof(n)\} >$[5]

and therefore, the above specification answers *no* to a query $\exists x\ (course(x) \& \neg H(x))$ and *unknown* to a query $\exists x\ (prof(x) \& \neg H(x))$. Notice that in this example it is essential to allow for the possibility of unknown objects.

[5] Of course, the world view also contains Ha, Hb, Hm, Hn, $\neg Hc_1 \ldots \neg Hc_k$. From now on, in the descriptions of world views we will omit literals formed with H.

Let us now expand an informal specification of our database by the closed world assumptions for predicates *course* and $prof$. Closed world assumption for *course* says that there are no other courses except those mentioned in the database and can be formalized by a standard rule

$\neg course(x) \leftarrow \neg \mathrm{M}\ course(x)$.

Using this assumption we will be able to prove that a and b are the only courses taught in our department. In the case of predicate $prof$, however, this (informal) assumption is too strong – there may, after all, be some unknown professor not mentioned in the list. However, we want to be able to allow our database to conclude that no one known to the database is a professor unless so stated. For that we need a weaker form of closed world assumption, which will not be applicable to generic elements. This can easily be accomplished by the following rule

$\neg prof(x) \leftarrow H(x), \neg \mathrm{M}\ prof(x)$.

The k-world view of a resulting specification Π looks as follows:

$< k, \{c(a), c(b), \neg c(m), \neg c(n), \neg c(c_1) \ldots \neg c(c_k), p(m), p(n), \neg p(a), \neg p(b)\} >$

where c stands for *course* and p stands for $prof$. This allows us to conclude, say, that a is not a professor without concluding that there are no professors except m and n.

3.2 Unique name assumption.

The unique name assumption [12] is normally used in settings when one can assume that all the relevant information about the equality of individuals has been specified. In this case **all pairs of individuals not specified as identical are assumed to be different**. To express this assumption we follow the approach from [3] and introduce a new binary predicate symbol E which stands for *equality*. The specification consisting of the rules (1) – (5) can be viewed as the definition of E:

E_1. $E(x, x) \leftarrow$

E_2. $E(x, y) \leftarrow E(y, x)$

E_3. $E(x, y) \leftarrow E(x, z), E(z, y)$

E_4. $F(y_1, \ldots, y_n) \leftarrow E(x_1, y_1), \ldots, E(x_n, y_n), F(x_1 \ldots x_n)$

E_5. $\neg E(x_1, y_1)\ or \quad \ldots\ or \quad \neg E(x_n, y_n) \leftarrow F(x_1 \ldots x_n), \neg F(y_1 \ldots y_n)$

for every objective formula F.

Obviously axioms E_1 – E_5 added to a specification Π whose language does not contain E do not change the set of formulae entailed by Π.

The next proposition allows one to slightly simplify the equality rules above.

Proposition 3. Let Π be an epistemic specification containing rules E_1 – E_5 and let Π^* be obtained from Π by restricting E_4 and E_5 to literals. Then for every statement F, $\Pi \models F$ iff $\Pi^* \models F$.

With equality available in the language we can express the unique name assumption as

$UNA.\quad \neg E(x,y) \leftarrow \neg \mathrm{M} E(x,y),$

i.e. as the closed world assumption for E. Notice that in this form the assumption is rather strong and is applicable to both named and generic elements of the domain. Other versions of the closed world assumption can be expressed in a similar manner.

To better understand the use of equality axioms and UNA in commonsense reasoning let us consider the following example:

Example 7. Suppose that $\mathcal{L}$ is a language containing a list of names such as $mike$, $john$, $mary$, etc. and assume the unique name assumption for these names. Suppose also that our specification contains the following complete list of professors in a computer science department:

1. $prof(mike) \leftarrow$

2. $prof(john) \leftarrow$

To express the completeness of the list we will use the closed world assumption

3. $\neg prof(x) \leftarrow \neg \mathrm{M}\, prof(x)$

while the unique name assumption is represented by axioms E_1 – E_4 and UNA.

Let us denote the specification E_1 – E_4, UNA, (1)–(3) by Π^0. For any $k \geq 0$ the world view of Π^0 is $\{prof(mike), prof(john)\}$ united with the set of literals of the form $\neg prof(c)$ where c is a constant (named or generic) of the corresponding language $\mathcal{L}_k$ different from $mike$ and $john$ and with the set of all literals of the form $E(a,a)$ and $\neg E(a,b)$ where a and b are pairs of constants from $\mathcal{L}_k$ such that a is not identical to b.

Let us now assume that Mike also goes by another name, say, Misha. This information can be coded in our system as

4. $E(mike, misha) \leftarrow$

The world view of the specification $\Pi^1 = \Pi^0 \cup (4)$ is obtained from the world view of Π^0 by replacing $\neg E(misha, mike)$, $\neg E(mike, misha)$, and $\neg prof(misha)$ by $E(misha, mike)$, $E(mike, misha)$ and $prof(misha)$ respectively. Therefore, the new information about equality allowes us to withdraw our previous conclusion about Misha's position in the department. Of course we still will be able to prove $\neg prof(mary)$, etc.

The absence of the unique name assumption makes our reasoning substantially more complicated. First, in the absence of complete information about equality we need to modify our formulation of the closed world assumption. A refined version of the assumption says that *anyone different from people included in the list of professors is not a professor.* To formalize this version let us introduce a new unary predicate symbol *could_be_prof* and define it as follows:

3a. $could\_be\_prof(x) \leftarrow prof(y), not\ \neg E(x,y)$

A new form of the closed world assumption looks as follows:

3b. $\neg prof(x) \leftarrow \neg \mathrm{M}\ could\_be\_prof(x)$.

To better understand these new axioms let us consider a specification Π^2 obtained from Π^1 by removing UNA and (3) and adding the rules (3a), (3b) and the new rules

5. $\neg E(greg, misha) \leftarrow$

6. $\neg E(greg, john) \leftarrow$.

Obviously, due to the incompleteness of information about equality, the new specification is no longer able to conclude $\neg E(mike, john)$ and $\neg prof(mary)$ but is still capable of inferring $\neg prof(greg)$.

Finally, to demonstrate the use of the rule E_5 let us consider Π^3 obtained from Π^2 by removing (3a), (3b) and adding E_5 and

7. $\neg\ prof(mary) \leftarrow$.

Let $\neg E(mary, misha)$ be a query to this specification. It is easy to see that (1), (4) and E_4 imply $prof(misha)$ which, together with (7) and E_5 implies $\neg E(mary, misha)$. It is obvious that the use of (3a) and (3b) and, especially, E_5 substantially increases the complexity of the reasoning process. It may be interesting to look for classes of specifications and/or queries for which E_5 is redundant.

3.3 Anonymous exceptions to defaults

Let us now consider a classical flying birds example [10], in which we are told that penguins are birds that do not fly, that birds normally fly, and that Tweety is a bird. The example served as a testing ground for various nonmonotonic formalisms. Many formalizations of this example can be found in the literature, but apparently none can be considered fully adequate. For instance, a natural circumscriptive formalization minimizing the set of nonflying birds allowing other predicates to vary implies that there are no penguins, etc. An interesting discussion of this and related problems can be found in [2].[6] The example can be easily coded in the language of epistemic

[6]It is important to notice that some of the criticism from [2] is based on the understanding of nonmonotonic reasoning as " reasoning that can reach conclusions that are

specifications. One possible way of doing it is given by axioms (1) – (5) below.

1. $Fx \leftarrow Bx, not\ ab(f,b,x), not\ \neg Fx$

2. $Bx \leftarrow Px$

3. $\neg Fx \leftarrow Px$

4. $\neg Px \leftarrow Fx$

5. $Bt \leftarrow$

It is easy to see that the k-world view of this specification is

$\{< k, \{Bt,\ Ft,\ \neg Pt\} >\}$

Therefore, Tweety is a flying bird, not a penguin, and the answer to a query $Q = \exists x Px$ is *unknown*. If we were to expand the above rules by

6. $Po \leftarrow$

where o stands for Opus, then Tweety would still fly, while Opus wouldn't and the answer to Q would be *yes*. The situation changes, however, if, instead of expanding the original specification by $Po \leftarrow$, we expand it by a rule

7. $\exists x Px \leftarrow$.

The world views of the resulting theory (1) – (5), (7) are

$\{\{< 0, \{Bt,\ Pt,\ \neg Ft\} >\}\}$

$\{\{< 1, \{Bt,\ Pt,\ \neg Ft\} >\}\{< 1, \{Bt,\ Ft,\ \neg Pt,\ Pc_1,\ Bc_1,\ \neg Fc_1\} >\}\}$

...

and therefore T no longer entails Ft. Unlike Etherington *et al.* we do not view this as counterintuitive. The new axiom certainly gives us a reason to suspect that Tweety can be a penguin and therefore correctly blocks the application of the default. To conclude Ft we should be able to simulate the following informal reasoning: "The existing penguin is apparently neither Tweety nor any other named individual but just a generic, unnamed penguin. Hence, Tweety is a flying bird and not a penguin." To do that, we should find reasoning principles justifying the first step of this argument. The first natural candidate is the open domain assumption stating existence of generic elements:

---

not *strictly* entailed by what is known". This is different from the epistemic view of this paper, according to which all conclusions of epistemic theory must be "strictly" entailed. From this standpoint some of the conclusions viewed as unintuitive [2] are to be expected and blamed on inadequate collection of axioms formalizing the problem and not on the formalism itself. Some of such problems can be avoided by expanding the original theories by new axioms describing properties of the world and a reasoner, and/or by finding more suitable translation of natural language sentences in the language of a given formalism. Others, such as the lottery paradox, depend on the size of domain and may require a somewhat different formalism.

$\exists x\ \neg Hx \leftarrow$

which will eliminate the 0–world view. This is a necessary step but we need something much stronger – existence of unnamed penguins. This may be achieved by the following default: "normally, if there are penguins then there are generic penguins". The default can be written as follows:

8. $\exists x\ (penguin(x)\ \&\ \neg H(x)) \leftarrow \exists x\ penguin(x), not\ closed(\_penguin)$

where $\_penguin$ is a new constant symbol. The specification (1) – (5), (7) – (8) has the world views:

$\{< 1, \{Bt,\ Ft,\ \neg Pt,\ Pc_1,\ Bc_1,\ \neg Fc_1\} >\}$

$\{< 2, \{Bt, Ft, \neg Pt, Pc_1, Bc_1, \neg Fc_1\} >, < 2, \{Bt, Ft, \neg Pt, Pc_2, Bc_2, \neg Fc_2\} >$
$\ldots$

and hence entails Ft and $\neg Pt$.

Suppose now we learned that there are no unnamed penguins, i.e. the domain of penguins is closed. This can be expressed by

9. $\neg penguin(x) \leftarrow \neg H(x)$

which conflicts with (7) and (8). To prevent a conflict we need a cancellation axiom

10. $closed(\_penguin) \leftarrow$

The new specification (1) – (5), (7) – (10) implies neither Ft nor Pt which seems to be the right conclusion.

Even though rule (8) is powerful enough for the above example it does not allow us to conclude that Tweety flies if we know about the existence of two anonymous but different penguins. The following more powerful rule will do the trick.

Let $diff\_peng(x, y)$ be a shorthand for a formula
$penguin(x)\ \&\ penguin(y)\ \&\neg E(x, y)$
expressing the fact that x and y are different penguins.

11. $\exists x, y\ diff\_peng(x, y)\ \&\neg H(x)\ \&\ \neg H(y)) \leftarrow \exists x, y\ diff\_peng(x),$
$not\ closed(\_penguin)$

The same method can be used to express that there typically are n generic penguins where n depends on a particular domain of discourse.

4 Relation to other work.

The fact that the incorporation of the domain–closure assumption in the semantics of logic program can cause some unintended consequences had been known for a long time. This is the case even for positive logic programs, i.e. programs of the form

$p_0 \leftarrow p_1, \ldots, p_m$

where p's are atoms. To illustrate this point let us consider the following simple example from the literature:

Example 8. Consider a positive logic program:

$Pa \leftarrow$

and a query $Q = \forall x Px$.

Under the domain–closure assumption the semantics of this program is given by its least Herbrand model [1], and hence Π's answer to a query Q will be *yes*. However, if we add to Π an apparently unrelated fact Rb, the answer of the new program Π^* to the same query Q becomes *no*. This lack of modularity, the surprising ability of a program to entail positive facts not entailed by the corresponding classical theory, etc. were recognized as a problem of the least Herbrand model semantics. T. Przymusinski in [30] termed the above problem the *universal query problem* and suggested as a solution the semantics of logic programs based on arbitrary (not necessarily Herbrand) minimal models. This allows us to avoid the universal query problem – under proper definition of an answer to a query both Π and Π^* answer *unknown* to Q. At the same time the semantics from [11] do not diverge too far from the least Herbrand model semantics. In fact, these two semantics are equivalent for existential queries [7]. [7] Our paper can be viewed as an extension of the approach from [11] to epistemic specifications. It is worth noting that even in the language of extended logic programs the presence of classical negation stresses the non–classical character of the connective $\leftarrow$ and forces us to abandon the classical notion of model as the basis for the semantics of logic program. This leads to some technical complications but at the same time allows us to define a predicate H which, together with the use of the rules with empty heads is the basic tool for formalization of the domain–closure and other assumptions.

[9] is another recent paper closely related to the subject of our work. It suggests a variant of Reiter's default logic aimed at formalization of reasoning about domains without domain–closure assumption. In this paper V. Lifschitz introduces the notion of F–consequence of a default theory which can easily be adapted to epistemic specifications. The corresponding consequence relation is "ideologically" similar to ours. There are, however, some important technical differences between the two. Consider, for instance an extended logic program Π:

[7] T. Przymusinski's approach is not limited to positive programs. In [11] it is extended to perfect model semantics, etc. Another solution of universal query problem is suggested in [14]. It is based on the assumption that the language of any logic program contains infinitely many constants not appearing in it explicitly. Under this semantics, both programs Π and Π^* answer *no* to the query Q, which, in a sense, amounts to preferring open domains over the closed ones. Such a preference appears somewhat arbitrary. Unless open or closed domain assumptions are stated explicitly, *unknown* seems to be more intuitive answer to Q.

$p(a) \leftarrow$

$\neg p(b) \leftarrow not\ p(b)$

It is easy to see that $\neg p(b)$ is a consequence of Π according to our semantics while, as mentioned in [9], it is not an F–consequence of Π. The difference seems to be primarily in the treatment of the relationship between defaults and equality. The application of the default (represented by the second rule) is prohibited in [9] because a can be equal to b while in our case the default is applied, which allows us to derive that a and b are different.

5 Conclusions

In this paper we modified the semantics of epistemic specifications to allow reasoning in the absence of domain–closure assumption. By way of example we demonstrated that this semantics together with the introduction of a predicate H for "named elements of the domain", simple equality theory and the use of rules with empty conclusions, allows us to express subtle forms of various domain assumptions. We also showed how the existence of unknown elements can be used to solve the problem of representing anonymous extensions to defaults. This, together with previous work in [6], [4], etc. shows the power of logic programming based formalisms as a tool for knowledge representation.

6 Acknowledgments

We would like to thank Vladimir Lifschitz, Tom Costello and Bonnie Traylor for useful comments on the form and content of this paper. The first author was supported in part by NSF grants CDA-9015006 and IRI-9101078.

References

[1] Maarten van Emden and Robert Kowalski. The semantics of predicate logic as a programming language. *Journal of the ACM*, 23(4):733–742, 1976.

[2] David Etherington, Sarit Kraus, and David Perlis. Nonmonotonicity and the scope of reasoning. *Artificial Intelligence*, 52(3):221–261, 1991.

[3] Michael Gelfond. Epistemic semantics for disjunctive databases. Preprint, ILPS Workshop on Disjunctive Logic Programs, San Diego, Ca., 1991.

[4] Michael Gelfond. Strong introspection. In *Proc. AAAI-91*, pages 386–391, 1991.

[5] Michael Gelfond and Vladimir Lifschitz. Logic programs with classical negation. In David Warren and Peter Szeredi, editors, *Logic Programming: Proc. of the Seventh Int'l Conf.*, pages 579–597, 1990.

[6] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, pages 365–387, 1991.

[7] Michael Gelfond, Halina Przymusinska, and Teodor Przymusinski. On the relationship between cwa, minimal model, and minimal herbrand model semantics. *International Journal of Intelligent Systems*, 5(5):549–565, 1990.

[8] Hector Levesque. Making believers out of computers. *Artificial Intelligence*, 30:81 – 108, 1986.

[9] Vladimir Lifschitz. On open defaults. In John Lloyd, editor, *Computational Logic: Symposium Proceedings*, pages 80–95. Springer, 1990.

[10] John McCarthy. Applications of circumscription to formalizing common sense knowledge. *Artificial Intelligence*, 26(3):89–116, 1986.

[11] Teodor Przymusinski. On the declarative and procedural semantics of logic programs. *Journal of Automated Reasoning*, 5:167–205, 1989.

[12] Raymond Reiter. On closed world data bases. In Herve Gallaire and Jack Minker, editors, *Logic and Data Bases*, pages 119–140. Plenum Press, New York, 1978.

[13] Raymond Reiter. Equality and domain closure in first-order databases. *JACM*, 27:235–249, 1980.

[14] Kenneth Ross. A procedural semantics for well founded negation in logic programming. In *Proc. of the eighth Symposium on Principles of Database Systems*, pages 22–34, 1989.