Strong Introspection*

Michael Gelfond
Computer Science Department
University of Texas at El Paso
El Paso, Texas 79968
cv00@utep.bitnet

Abstract

The purpose of this paper is to expand the syntax and semantics of logic programs and deductive databases to allow for the correct representation of incomplete information in the presence of multiple extensions. The language of logic programs with classical negation, epistemic disjunction, and negation by failure is further expanded by a new modal operator K (where for the set of rules T and formula F, KF stands for "F is known to a reasoner with a set of premises T"). Theories containing such an operator will be called strongly introspective. We will define the semantics of such theories (which expands the semantics of deductive databases from [Gelfond and Lifschitz 1990b]) and demonstrate the applicability of strongly introspective theories to formalization of some forms of commonsense reasoning.

Introduction

In recent years a substantial amount of work was done to investigate the applicability of logic programming based formalisms to knowledge representation (for a good overview see [Kowalski 90]). The results are very promising but some substantial problems related to this approach still remain unsolved. One of such problems – inability to deal directly with incomplete information – was discussed in [Gelfond and Lifschitz 1990a]. The language of extended logic programs proposed there partially overcomes this limitation by distinquishing between two types of negation: "classical" (or "strong") negation $\neg$ (where $\neg F$ can be interpreted as "F is known to be false"), and negation as failure *not* (where *not* F can be interpreted as "F is not known to be true"). [1] The semantics of extended logic programs is based on the notion of answer sets—sets of literals which can be viewed as theories satisfying the corresponding program. For extended programs without classical negation their answer sets coincide with stable models from [Gelfond and Lifschitz, 1988]. In [Gelfond and Lifschitz, 1990a] we considered primarily well-behaved extended logic programs, i.e. extended programs with unique consistent answer sets. The answer such a program is supposed to return to a ground query Q is *yes*, *no*, or *unknown*, depending on whether the answer set contains Q, $\neg Q$, or neither. The existence of several answer sets indicate that the corresponding program P has several possible interpretations, i.e. it is possible for a rational reasoner to construct several theories satisfying P. Such a multiplicity becomes a norm rather then exception if the notion of extended logic program and its answer set semantics is expanded to disjunctive databases (see [Gelfond and Lifschitz 1990b], [Gelfond 1990]) — collections of rules of the form:

$$A_1 \ or \ \ldots \ or A_n \leftarrow B_1, ..., B_m, not\ C_1, \ldots, not\ C_k$$

where A's, B's, and C's are atoms p or their "classical" negations $\neg p$. (Notice the use of symbol *or* instead of classical $\vee$. The meaning of a connective *or*, called epistemic disjunction, is given by the semantics of disjunctive databases and differ from that of $\vee$. The meaning of a formula $A \vee B$ is "A is true or B is true" while a rule $A \ or B \leftarrow$ is interpreted epistemically and means "belief A or belief B".) The answer such a database T returns to a ground query Q is *yes* if Q belongs to all answer sets of T, *no* if all answer sets contain $\neg Q$, or *unknown* otherwise. (The last answer can be split into several more informative answers but above alternatives are sufficient for the purpose of this paper). In [Gelfond and Lifschitz 1990] we argued that for well-behaved programs the presence of two types of negation allows one to deal in a natural and convenient way with incomplete information. This, however, is no longer the case if the corresponding programs (or deductive databases) are not well-behaved. The purpose of this paper is to expand the notions of extended logic programs and deductive databases to allow for the correct representation of incomplete information in the presence of multiple answer sets.

We will start by demonstrating the problem using the following example from [Gelfond and Lifschitz 1990]:

*This research was partially supported by NSF grant #IRI89-06516

[1] A similar approach was independently developed and investigated in [Pearce and Wagner, 1989], [Wagner, 1990].

Example 1 Considered a collection of rules

1. $Eligible(x) \leftarrow HighGPA(x)$,
2. $Eligible(x) \leftarrow Minority(x), FairGPA(x)$,
3. $\neg Eligible(x) \leftarrow \neg FairGPA(x)$,
4. $Interview(x) \leftarrow not\ Eligible(x),$
   $not\ \neg Eligible(x)$

used by a certain college for awarding scholarships to its students. The first three rules are self explanatory while the fourth rule can be viewed as a formalization of the statement:

(*) "The students whose eligibility is not determined by the first three rules should be interviewed by the scholarship committee".

In its epistemic form the rule says : $Interview(x)$, if neither $Eligible(x)$ nor $\neg Eligible(x)$ is known. We assume that this program is to be used in conjunction with a database DB consisting of literals specifying values of the predicates *Minority*, *HighGPA*, *FairGPA*. Consider, for instance, DB consisting of the following two facts about one of the students:

5. $FairGPA(ann) \leftarrow$.
6. $\neg HighGPA(ann) \leftarrow$.

(Notice that DB contains no information about the minority status of Ann). Intuitively it is easy to see that rules (1)–(6) allow us to conclude neither $Eligible(ann)$ nor $\neg Eligible(ann)$, therefore eligibility of Ann for the scholarship is undetermined and, by rule (4), she must be interviewed. Formally this argument is reflected by the fact that program T_1 consisting of rules (1)–(6) has exactly one answer set:

$$\{FairGPA(ann), \neg HighGPA(ann), Interview(ann)\}.$$

The situation changes significantly if disjunctive information about students is allowed to be represented in the database. Suppose, for instance, that we need to augment rules (1)–(3) by the following information:

(**) Mike's GPA is fair or high.

There are several possible ways to represent this information. The most natural one seems to be to use the language and semantics of deductive databases from [Gelfond and Lifschitz, 1990b]. The corresponding deductive database T_2 consists of rules (1)–(3) augmented by the disjunction

7. $FairGPA(mike)\ or\ HighGPA(mike) \leftarrow$

T_2 has two answer sets:

$$A_1 = \{HighGPA(mike), Eligible(mike)\}$$

and

$$A_2 = \{FairGPA(mike)\},$$

and therefore the reasoner modeled by T_2 does not have enough information to establish Mike's eligibility for the scholarship (i.e. answer to $Eligible(mike)$ is *unknown*). If we now expand this theory by (*) we expect the new theory T_3 to be able to answer *yes* to a query $Interview(Mike)$. It is easy to see however that if (*) is represented by (4) this goal is not achieved. The resulting theory T_3 consisting of (1)–(4) and (7) has two answer sets

$$A_3 = \{HighGPA(mike), Eligible(mike)\}$$

$$A_4 = \{FairGPA(mike), Interview(mike)\}$$

and therefore the answer to query $Interview(mike)$ is *unknown*. The reason of course is that (4) is too weak to represent (*). The informal argument we are trying to capture goes something like this: theory T_3 answers neither *yes* nor *no* to the query $Interview(mike)$. Therefore, answer to this question is undetermined, and, by (*), Mike should be interviewed. To formalize this argument our system should have a more powerful introspective ability then the one captured by the notion of answer sets from [Gelfond and Lifschitz 1990b]. Roughly speaking instead of looking at only one possible set of beliefs sanctioned by T it should be able to look at all such sets.

Remark. The situation will not change if (**) is represented by modeling disjunctions in the language of logic programs. For instance, replacing (7) by two rules

$$FairGPA(Mike) \leftarrow not\ HighGPA(Mike)$$

$$HighGPA(Mike) \leftarrow not\ FairGPA(Mike)$$

will not change answer sets of the program.

In this paper we extend the syntax of deductive databases from [Gelfond and Lifschitz 1990b] in two directions. Firstly, following [Lloyd and Topor 1984], and [Wagner 1990] we allow the rules to contain other types of formulae except literals. Secondly, and more importantly, we expand the language by a modal operator K (where for any set of rules T and formula F, KF stands for "F is known to a reasoner with a given set of premises T"). Theories containing such an operator will be called strongly introspective. We will define the semantics of such theories (which expands the semantics of deductive databases from [Gelfond and Lifschitz 1990b] and demonstrate the applicability of strongly introspective theories for formalization of some forms of commonsense reasoning.

Definitions

Let us consider a language $\mathcal{L}_0$ consisting of predicate symbols $p, q, \ldots$, object variables, function symbols, logical connectives &, $\neg$, *not*, $\exists$, and the modal operator K. Formulae of $\mathcal{L}_0$ will be defined in the usual way. Formulae not containing modal operators will be called objective formulae while those starting with K will be called subjective. Formulae of the form $p(a)$ will be called objective atoms. By objective literals we will

mean objective atoms $p(a)$ and their strong negations $\neg p(a)$. Literals not containing variables will be called ground. The set of all ground literals will be denoted by *Lit*. Formulae not containing free variables will be called statements.

Let us consider a collection $A = \{A_i\}$ of sets of ground objective 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 $\mathcal{L}_0$ w.r.t. a pair $M =< A, W >$.

Definition 1

$M \models p(a)$ *iff* $p(a) \in W$.
$M \models \mathrm{K}F$ *iff* $< A, A_k > \models F$ *for every* 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 *such that* $M \models F(t)$
$M \models \neg F$ *iff* $M =| F$
$M \models$ *not* F *iff* $M \not\models F$
$M =| p(a)$ *iff* $\neg p(a) \in W$
$M =| \mathrm{K}F$ *iff* $< A, A_k > =| F$ *for some* A_k *from* A
$M =| F\&G$ *iff* $M =| F$ *or* $M =| G$
$M =| \exists x F$ *iff for every ground term* t, $M =| F(t)$
$M =| \neg F$ *iff* $M \models F$
$M =|$ *not* F *iff* $M \models F$

It is easy to see that according to this definition 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 $A \models F$ ($A =| F$).

In our further discussion we will expand language $\mathcal{L}_0$ by the connectives *or* and $\forall$ and a modal operator M (where MF will be read as "F may be believed") defined as follows:

MF iff $\neg \mathrm{K} \neg F$
F *or* G iff $\neg(\neg F \,\&\, \neg G)$
$\forall x F$ iff $\neg \exists x \neg F$

The language and the satisfiability relation described above together with the notion of a rule from logic programming will be used to provide a specification of a reasoner with the desired properties. (This view on the role of logic in nonmonotonic reasoning seems to be similar to the one advocated by H. Levesque in [Levesque 90]). The formal notion of such a specification is captured by the following definitions:

Definition 2 *By an epistemic specification we will mean a collection of rules of the form*

$$F \leftarrow G_1, \ldots, G_m$$

where F *is objective and* G*'s are arbitrary formulae.*

Now we will define a collection A of sets of literals (which can be viewed as vivid theories in terminology from [Levesque 86]) satisfying an epistemic specification T. We will call such a collection a *world view* of T and its elements belief sets of T. The precise definition of these notions will be given in several steps:

Definition 3 *Let* T_0 *be an epistemic specification consisting of rules of the form*

$$F \leftarrow$$

A set W *of literals is called a* **belief set** *of* T_0 *iff* W *is a minimal set with a property* $W \models F$ *for every rule from* T_0. *If* W *contains a pair of complementary literals then* $W = Lit$.

Example 2 Let T consist of clauses:

1. $p(a)$ *or* $p(b) \leftarrow$
2. $\neg p(b) \leftarrow$
3. *not* $q(b) \leftarrow$
4. $r(a)$ *or* $\neg r(b) \leftarrow$
5. $\exists x q(x) \leftarrow$

It is easy to see that T has two belief sets:

$\{p(a), \neg p(b),\ q(a),\ r(a)\}$ $\{p(a), \neg p(b),\ q(a), \neg r(b)\}$

Definition 4 *Let* T *be an arbitrary epistemic specification,* W *be a set of literals in the language of* T *and* **A** *be a collection of sets of literals in this language. For every* $M =< A, W >$ *by* T_M *we will denote the epistemic specification obtained from* T *by:*

1. *Removing from the premises of rules of* T *all formulae* G *such that* $M \models G$.
2. *Removing all remaining rules with non-empty premises.*

Then **A** *will be called a world view of* T *iff* $A = \{W : W$ *is a belief set of* $T_M\}$ *where* $M =< A, W >$.

Example 3 Let T consist of the formulae

1. $P(a)$ *or* $P(b) \leftarrow \exists x Q(x)$, *not* $Q(d)$
2. $Q(c) \leftarrow$

It is easy to see that this specification has a unique world view consisting of two belief sets $\{Q(c),\ P(a)\}$ and $\{Q(c),\ P(b)\}$.

Example 4 Let T consist of the formulae

1. Pa *or* $Pb \leftarrow$
2. $Pc \leftarrow$
3. $Qd \leftarrow$
4. $\neg Px \leftarrow$ *not* MPx

It is easy to show that
$A = \{\ \{Qd,\ Pa,\ Pc, \neg Pd\},\ \{Qd,\ Pb,\ Pc, \neg Pd\}\}$
is the only world view of T.

Example 5 Let T consist of the formulae

1. $Pa \leftarrow$
2. $Qb \ or \ Qc \leftarrow$
3. $Rx \leftarrow not \ \mathrm{K}Qx$
4. $Sx \leftarrow not \ \mathrm{M}Qx$

The only world view of T is
$A = \{\{\text{Pa, Qb, Ra, Rb, Rc, Sa}\},$
$\{\text{Pa, Qc, Ra, Rb, Rc, Sa}\}\}.$

Example 6 Let $T = \{p \leftarrow not \ \mathrm{K}p\}$. It is easy to see that T does not have a world view.

Example 7 Let
$T = \{p \leftarrow not \ \mathrm{M}\ q, q \leftarrow not \ \mathrm{M}\ p\}$. This specification is satisfied by two world views: $A_1 = \{\{q\}\}$ and $A_2 = \{\{p\}\}$.

Definition 5 *We will say that a world view of epistemic specification T is consistent if it does not contain a belief set consisting of all literals.*

Definition 6 *We will say that epistemic specification is consistent if it has at least one consistent world view.*

Example 8 Let T consist of formulae p and $\neg p$. It is easy to see that theory T is inconsistent. Another inconsistent specification is given in Example 6.

Definition 7 *A specification is called* **well-defined** *if it has exactly one consistent world view.*

From now on we will only consider well-defined specifications.

Definition 8 *Let T be a specification and $A = \{A_i\}$ be its world view. A formula F is* **true** *in T ($T \models F$) iff $< A, A_i > \models F$ for every A_i from A.*

This definition can be used to define the range of possible answers to a query Q. For the purpose of this paper we will limit ourself to the simple case when answer to Q is *yes* if $T \models Q$, *no* if $T \models \neg Q$, and *unknown* otherwise.

The following two Propositions establish the relationship between deductive databases and epistemic specifications.

Proposition 1. Let T be an epistemic specification consisting of clauses of the form
$F \leftarrow G_1, \ldots, G_m, not \ E_{m+1}, \ldots, not \ E_k.$
Then

1. If F, G's and E's are atoms (i.e. T is a general logic program) then A is a world view of T iff A is the set of all stable models of T.
2. If G's, and E's are objective literals and F is a disjunction of objective literals (i.e. T is a deductive database) then A is a world view of T iff A is the set of all answer sets of T.

Proposition 2. Let T be a well-defined specification with the world view A. Then A is a singleton iff $W \in A$ is the unique answer set of the theory T' obtained from T by deleting modal operators K and M.

Applications

In this section we will discuss several applications of epistemic specifications to formalization of commonsense reasoning.

(a) Representing the Unknown.

We will start with demonstrating how statements of the form "unknown p" can be represented by strongly introspective formulae. We suggest to represent formulae of this form as a conjunction of formulae *not* K p and *not* K $\neg p$. Let us go back to Example 1 from the introduction to illustrate this point.

Example 1 revisited. Let us consider the theory consisting of rules (1)– (3) and (7) from Example 1. To obtain the proper formalization of (*) we will just replace statement (4) by

4'. $Interview(x) \leftarrow not \ \mathrm{K}\ Eligible(x),$
$not \ \mathrm{K}\ \neg Eligible(x)$

which corresponds closely to the intuitive meaning of (*). It is easy to check that the theory T consisting of rules (1)–(3), 4', and (7) has the world view
$A = \{A_1, A_2\}$ where

$A_1 = \{HighGPA(Mike), Eligible(Mike), Interview(Mike)\}$

$A_2 = \{FairGPA(Mike), Interview(Mike)\}$

Therefore T answers *unknown* to the query $Eligible(Mike)$ and *yes* to the query $Interview(Mike)$ which is the intended behaviour of the system.

(b) Closed World Assumption.

Now we will illustrate how strong introspection can be used to represent the Closed World Assumption (CWA) of [Reiter 1978] for non-Horn databases (i.e. databases containing disjunctions). The question of formalizing this assumption has been extensively studed by various authors (for a good review see [Przymusinska and Przymusinski 1990]). [Minker 1982] gives perhaps the most widely known form of the CWA for non-Horn databases. As was noticed in [Ross and Topor 1988], this assumption tends to interpret disjunction as exclusive. Some attempts to remedy this problem can be found in [Chan 89], [Sakama 89] and [Gelfond 90]. Full discussion of these and other approaches is beyond the scope of this paper. Instead, we suggest a formalization of CWA that we believe is more general. We will start with the following example:

Example 9. [Chan 89]. Suppose we are given the following information:

(*) "If suspect is violent and is a psychopath then the suspect is extremely dangerous. This is not the case if the suspect is not violent or not a psychopath"

which is used in conjunction with a database DB consisting of literals specifying values of the predicates

violent and *psychopath*. Let us also assume that DB contains complete positive information about these predicates, i.e. ground atoms with predicate symbols *violent* and *psychopath* assumed to be false if there is no evidence to the contrary. This statement can be viewed as an informal description of Reiter's Closed World Assumption (CWA).
The information from (*) can be easily expressed by three statements

1. $dangerous(x) \leftarrow violent(x), psychopath(x)$
2. $\neg dangerous(x) \leftarrow \neg violent(x)$
3. $\neg dangerous(x) \leftarrow \neg psychopath(x)$

Formalization of CWA is somewhat more problematic. In [Gelfond and Lifschitz 1990] we suggested to express CWA for a predicate $P(x)$ by the rule

$$\neg P(x) \leftarrow not\ P(x)$$

As expected this formalization works nicely for well-defined extended programs but is not suitable in the general case. To see the problem let us apply this idea to our example. CWA for predicates *violent* and *psychopath* will look as follows:

4. $\neg violent(x) \leftarrow not\ violent(x)$.
5. $\neg psychopath(x) \leftarrow not\ psychopath(x)$

Suppose that our DB contains the following information:

6. $violent(john) \leftarrow$,
7. $violent(mike) \leftarrow$,
8. $psychopath(mike) \leftarrow$.

It is easy to check that the theory T_1 consisting of clauses (1)–(8) is well-defined and has exactly one belief set A_0:

$$\{violent(john),\ violent(mike),$$
$$psychopath(mike),\ \neg psychopath(john),$$
$$\neg dangerous(john),\ dangerous(mike)\}$$

which properly reflects our intuition. The situation changes when disjunctive information is allowed in DB. Consider, for instance, a statement

9. $violent(sam)$ *or* $psychopath(sam)$

and a specification T_2 consisting of clauses (1)–(5) and (9). Notice that (9) is not an exclusive disjunction and therefore T_2 does not seem to sanction the conclusion

10. $\neg dangerous(sam)$.

But it is easy to see that T_2 does imply (10) which seems to be overly optimistic. The problem is apparently caused by an incorrect formalization of CWA — the fact that $not\ violent(sam)$ is true in one of the belief sets of T_2 does not guarantee that, given T_2, a rational reasoner does not have a reason to belief $violent(sam)$. In the case of T_2 such reason may be given by the existence of a belief set containing $violent(sam)$. This consideration leads us to the better representation of CWA for a predicate P which is provided by the statement

$$\neg P(x) \leftarrow not\ \mathrm{M}P(x)$$

It is easy to see that for well-defined programs both formalizations of CWA coincide.

Let us now consider theory T_3 obtained from T_2 by replacing statements (4) and (5) by

4'. $\neg violent(x) \leftarrow not\ \mathrm{M}violent(x)$.
5'. $\neg psychopath(x) \leftarrow not\ \mathrm{M}psychopath(x)$

The resulting theory has exactly one world view A consisting of two belief sets $A_1 = A_0 \cup \{violent(sam)\}$ and $A_2 = A_0 \cup \{psychopath\ (sam)\}$. T_3 implies neither (10) nor its negation and therefore answer to the query $dangerous(sam)$ is unknown.

(c) Integrity Constraints

We will finish our discussion of applicability of strong introspection to formalization of commonsense reasoning by an example demonstrating the utility of strong introspection for expressing integrity constraints.

Example 10. Let us assume that we are given the specification for a departmental database T:
(a) T should contain lists of professors, courses and teaching assignments for a CS department. Let us first assume that the department consists of professors named Sam and John and offers two courses: Pascal and Assembler, taught by Sam and John respectively.
(b) The above lists contain all the relevant positive information about the department known to us at a time.
(c) T must satisfy the following integrity constraint: Pascal is taught by at least one professor.

Part (a) of the specification is formalized as follows:

1. $prof(sam)\ \&\ prof(john) \leftarrow$
2. $class(pascal)\ \&\ class(assembler) \leftarrow$
3. $teach(sam, pascal)\ \&\ teach(john, assembler) \leftarrow$

Part (b) of the specification can be viewed as the Closed World Assumption and represented as

4. $\neg P(x) \leftarrow not\ \mathrm{M}P(x)$

for every predicate symbol P.

Formalization of part (c) seems to be the less obvious task. The main difficulty is related to the lack of universally excepted interpretation of the meaning and role of integrity constraints in knowledge representation. In this paper we will adopt the view on integrity constraints recently suggested by Reiter in [Reiter 1990]. According to Reiter an integrity constraint IC is a statement about the content of the knowledge

base T (as opposed to IC being a statement about the world). T satisfies IC iff the answer to IC when viewed as a query to T is *yes* . A simple analysis of clause (c) from this standpoint shows that (c) can be interpreted in two different ways:

IC_1 : K $\exists p(prof(p)$ & $teach(p, pascal))$
or

IC_2 : $\exists p$ K $(prof(p)$ & $teach(p, pascal))$

The first one says that the database knows that Pascal is taught by some professor (whose name can be unknown to the database), while the second one means that there is a person known to the database to be a professor teaching Pascal. It is easy to see that T_1 consisting of rules (1)–(3) satisfies both integrity constraints.
If however, we consider T_2 obtained from T_1 by replacing rule 2 by

2'. $teach(sam, pascal)$ *or* $teach(john, pascal)$

the situation changes.

It is easy to check that T_2 satisfies IC_1 but not IC_2. This is of course the intended result since this time the database does not know what professor will teach Pascal but knows that Pascal will be taught.

Remark. Even though the approach to formalization of integrity constraint suggested in this paper is similar to the one of Reiter there are some important differences: Reiter views a knowledge base as a first-order theory and a query as a statement of Levesque's modal logic (called KFOPCE). In our case knowledge base and queries are both epistemic formulae while the underlying logic is nonmonotonic.

Acknowledgments

I am grateful to Vladimir Lifschitz and Halina Przymusinska for comments on the first draft of this paper.

References

[Chan 1989] E. Chan, A Possible World Semantics for Non-Horn Databases, Research Report CS-89-47, University of Waterloo, Waterloo, Ontario, Canada, 1989

[Gelfond 1990] M. Gelfond, Epistemic Approach to Formalization of Commonsense Reasoning, Research Report, University of Texas at El Paso, 1990

[Gelfond and Lifschitz 1988] M. Gelfond and V. Lifschitz, The Stable Model Semantics for Logic Programs, Proceedings of the Fifth International Conference and Symposium on Logic Programming, (Kowalski, R.A. and Bowen, K.A. editors), vol. 2, pp. 1070-1080, 1988

[Gelfond and Lifschitz 1990a] M. Gelfond and V. Lifschitz, Logic Programs with Classical Negation, Proceedings of the Seventh International Conference on Logic Programming, MIT Press, pp. 579-597, 1990

[Gelfond and Lifschitz 1990b] M. Gelfond, V. Lifschitz, Classical Negation in Logic Programs and Deductive Databases, Research Report, University of Texas at El Paso and Stanford University, 1990

[Kowalski 90] R. A. Kowalski, Problems and Promises of Computational Logic, Proc. Symposium on Computational Logic, Springer-Verlag, 1990

[Levesque 1986] H. Levesque, Making Believers out of Computers, Artificial Intelligence 30, (1986) pp. 81-108

[Levesque 1990] H. Levesque, All I Know: A Study in Autoepistemic Logic, Artificial Intelligence 42, (1990) pp. 263-309

[Lloyd and Topor 1984] J. W. Lloyd and R. W. Topor, Making Prolog More Expressive, Journal of Logic Programming, 1984 : 3, pp. 225-240

[Minker 1982] J. Minker, On indefinite databases and closed world assumption. In Lecture Notes on Computer Science 138, pg. 292-38, 1982.

[Pearce and Wagner 1989] Pearce, G. Wagner, Reasoning with negative Information 1 - Strong Negation in Logic Programming, Technical Report, Gruppe fur Logic, Wissenstheorie und Information, Freie Universitat Berlin, 1989

[Przymusinska and Przymusinski 1990] H. Przymusinska, T. Przymusinski, Semantic Issues in Deductive Databases and Logic Programs, In R. Banerji, editor, Formal Techniques in Artificial Intelligence, pp. 321-367, North-Holland, Amsterdam, 1990

[Reiter 1978] R. Reiter, On Closed World Data Bases. In H. Gallaire and J. Minker, editors, Logic and Data Bases, pg 119-140, Plenum, New York, 1978.

[Reiter 1990] R. Reiter, On asking what a database knows, Proc. of Symposium on Computational Logic, Springer-Verlag, 1990

[Ross and Topor 1988] K.Ross, R. Topor, Inferring Negative Information from Disjunctive Databases, Journal of Automated Reasoning, (4:4), pp. 397-424, 1988

[Sakama 1989] C. Sakama, Possible Model Semantics for Disjunctive Databases, Proceedings of the first International Conference on Deductive and Object Oriented Databases, Kyoto, Japan, 1989

[Wagner 1990] G. Wagner, Logic Programming with Strong Negation and Inexact Predicates, 1990