Acta Cybernetica, Vol. 12, No. 1, Szeged, 1995

The Optimistic and Cautious Semantics for Inconsistent Knowledge Bases

John Grant [*] V.S. Subrahmanian[†]

Abstract

We develop two alternative semantics, based on maximal consistent subsets, for knowledge bases that (possibly) contain inconsistencies. The optimistic (resp. cautious) semantics correspond to entailment in some (resp. all) maximal consistent subsets. We develop a Kripke-style model theory corresponding to these two semantics. We further extend these semantics to the case when knowledge bases contain both explicit and nonmonotonic negation. Notions of stratification and stability are defined and studied for both semantics.

1 Introduction

Databases and knowledge bases may be inconsistent for various reasons. For example, during the construction of an expert system, we may consult many different experts. Each expert may provide us with a group of facts and rules which are individually consistent. However, when we coalesce the facts and rules provided by these different experts, inconsistency may arise. Such an inconsistency may be due to various factors such as a disagreement between experts, an error made by an expert, or a misunderstanding between experts. In any case, we may be forced to reason in the presence of an inconsistency. Classical logic is not adequate in this situation because a single inconsistency makes all possible statements true, thereby trivializing the whole knowledge base.

In a previous companion paper [8], we developed the so-called over-determined (or OD) semantics for reasoning in inconsistent knowledge bases. However, OD-semantics is not based on classical model theory: it allows models to make a statement and its negation both simultaneously true. The semantics we develop in this paper takes the meaning of an inconsistent knowledge base to be the set of maximal consistent subsets of the knowledge base. The optimistic semantics deduces all statements deducible from at least one maximal consistent subset. In the scenario involving many experts, the optimistic semantics accepts all statements

[*]Department of Computer and Information Sciences, Towson State University, Towson, MD 21204. E-mail: grant@midget.towson.edu.

[†]Department of Computer Science, Institute for Advanced Computer Studies, and Institute for Systems Research, University of Maryland, College Park, MD 20742. E-mail: vs@cs.umd.edu.

that at least one expert can deduce and possibly additional statements deducible from the knowledge of the experts that do not involve any inconsistencies. The cautious semantics deduces those statements that are deducible from all maximal consistent subsets. In the scenario involving many experts, the cautious semantics accepts all statements that every expert can deduce and possibly additional statements deducible from the knowledge of the experts that do not involve any inconsistencies.

The organization of this paper is as follows: Section 2 contains the basic notation and definitions. It also includes a specific example that motivates the semantics for inconsistent knowledge bases. Section 3 provides the definitions and basic results for optimistic and cautious entailment. In Section 4, a Kripke semantics is developed for optimistic and cautious entailment and a fixpoint operator is presented for optimistic entailment. In Section 5, stratification and stability are extended to our framework, and their relationship is investigated. Section 6 contains a summary and a discussion of related work.

2 Motivation and Example

We assume that the facts and rules of a knowledge base are expressed as clauses of the form

$$L_0 \leftarrow L_1 \,\&\ldots\&\, L_n$$

where each L_i, $0 \leq i \leq n$, is a literal (positive or negative). Initially, only classical negation ($\neg$) is used, but in Section 5, non-monotonic negation (**not**) is added. A clause of the above form is called a *generally Horn clause*. Note that a generally Horn clause allows a negative literal in the head of a clause. A knowledge base is represented in the form of a *generally Horn program* (GHP, for short), which is a set, possibly infinite, of generally Horn clauses. A logical language $\mathcal{L}$ generated by a finite number of constant, function and predicate symbols (and infinitely many variable symbols) is implicit in our setup. Throughout the paper, we consider $\neg\neg A$ to be synonymous with A, i.e. double negations are deleted. We use the notation $grd(P)$ to denote the set of all ground instances of clauses in P; in fact, usually we will assume that a generally Horn program is already in ground form.
As usual, the symbol $\models$ denotes semantic consequence, i.e. $P \models L$ means that L is a semantic logical consequence of P with respect to the semantics of classical two-valued logic. In the next section, we will introduce two new notions of entailment: $\vdash_\exists$ for the optimistic semantics, and $\vdash_\forall$ for the cautious semantics.
Next, we present a motivating scenario that exemplifies situations involving inconsistencies that arise in law enforcement agencies and in the judicial process:
Bill hosted a dinner at his house on Jan. 26, 1995. The party was attended by Al, Carl, Dick, Ed and Tom. Tom had to leave during dinner because his daughter had a medical emergency. After dinner, Bill went to the kitchen to prepare coffee. As he did not return, Dick and Ed went to the kitchen where they found Bill strangled to death. At the time of the crime:

- (F1) Tom was in the emergency room of a hospital. His presence was recorded by a surveillance camera belonging to hospital security.
- (F2) Bill was in the kitchen.
- (F3) Dick and Ed said they were talking in the living room.
- (F4) Al said he was alone in the bathroom.

- (F5) Carl said he was alone in the bathroom.

Furthermore,

- (F6) Bill's house has only one bathroom.
- (F7) Al had been guilty of embezzling money from Bill's accounting firm.
- (F8) Carl was having an affair with Bill's wife. Bill was an intensely jealous man.

Let us examine the story more carefully. First of all, the story contains a glaring contradiction: Al and Carl's stories conflict. This suggests that one of them is lying and we may further suspect that the person who is lying is the murderer. For lack of additional information, we are unable to determine which of them is actually the criminal. In this situation, the police may well decide to forget about Dick and Ed and look more closely at Al and Carl. More fantastic scenarios are also possible: Al and Carl may have been in cahoots and killed Bill and then both lied so that a convincing case could not be made against either of them. Alternatively, it is possible that everybody (except Tom) is lying: Al may have been in Bill's study trying to steal documentary evidence of his embezzlement, while Carl may have been in Bill's bedroom trying to get back his tie which he had left behind during one of his previous soirees with Bill's wife. While this was going on, Dick and Ed (both in cahoots) may have teamed up and killed Bill.
Whether one chooses to believe the above scenarios or not, one must admit that each of them is *possible*, though some are perhaps more probable than others.
However, whatever version we choose to believe, we would all be agreed that the general floor plan of Bill's house should be the same in all versions of the story. Likewise, the fact that Bill was strangled is true in all versions. In other words, in all versions of the story, certain facts are true. One may accept all these facts as being "certain" or established. The different versions of the story would tell us who to believe and who not to believe – in other words, they identify the suspects. Presumably, Tom would not be a suspect.
A formal logical description of the scenario is given in the Appendix. The cause of the inconsistency in the above example is the set of sentences $Cause = \{13, 14, 15, 16, 17\}$. Maximal consistent subsets may be obtained by dropping any one of these clauses.

3 Optimistic and Cautious Entailment

Suppose P is a GHP. We say that the success set of P, denoted $SS(P)$ is the set $\{L \mid L \text{ is a ground literal such that } P \models L\}$. Note that as P is a first order theory, $\models$ denotes standard semantical consequence in first order logic.

Example 3.1 Suppose P is the GHP:

$$p \leftarrow$$

$$\neg q \leftarrow p$$

$$r \leftarrow q$$

Then $SS(P) = \{p, \neg q\}$. Neither r nor $\neg r$ is in $SS(P)$. In particular, non-monotonic inference rules such as negation as failure and/or the closed world assumption are not used here because negation is represented explicitly.

A set $Q \subseteq P$ (where P is a GHP) is said to be *maximal consistent* iff Q is consistent, and there is no consistent program Q' such that $Q \subset Q' \subseteq P$.

Theorem 1 *Every GHP P has at least one maximal consistent subset.*

Proof. P has at least one consistent subset, viz. the empty set of clauses. Let $CONS(P)$ be the set of all consistent subsets of P. We show below that every ascending chain of elements in $CONS(P)$ has an upper bound in $CONS(P)$. The result then follows from Zorn's Lemma.
Suppose $S_1 \subseteq S_2 \subseteq S_3 \subseteq \cdots$ is an ascending sequence of members of $CONS(P)$, i.e. each S_i is a consistent subset of $CONS(P)$. Then $S = \bigcup_{i=1}^{\infty} S_i$ is an upper bound for this ascending sequence. Moreover, S is consistent, i.e. $S \in CONS(P)$. To see this, suppose S is not consistent. Then, by the Compactness Theorem, there is a finite subset $S' \subseteq S$ such that S' is inconsistent. Let $S' = \{\gamma_1, \ldots, \gamma_n\}$ for some integer n. Hence, for each $1 \leq i \leq n$, there is an integer, denoted α_i such that $\gamma_i \in S_{\alpha(i)}$. Let $\alpha = max\{\alpha_1 \ldots, \alpha_n\}$. Then $S' \subseteq S_\alpha$. Hence, as S' is inconsistent, S_α is also inconsistent, thus contradicting our assumption that each S_j, $j \geq 1$, is in $CONS(P)$. $\square$
Note that the above proof applies when P is any set of formulas, not just clauses. Furthermore, the proof applies even if P is an infinite set of formulas.

Example 3.2 Suppose P and Q are the two programs listed below:

P	Q
$p \leftarrow q$	$p \leftarrow r$
$\neg p \leftarrow \neg q$	$\neg p \leftarrow$

Here, $SS(P) = \emptyset$, while $SS(Q) = \{\neg p, \neg r\}$. Note that $Q \models \neg r$ because $(p \vee \neg r)$ and $\neg p$ yield $\neg r$ as a logical consequence.

Definition 3.1 Suppose P is a GHP, and F is a formula. We introduce[1] two new notions of entailment, denoted $\vdash_\exists, \vdash_\forall$ below:

1. $P \vdash_\exists F$ iff there is some maximal consistent subset $P' \subseteq P$ such that $P' \models F$.
2. $P \vdash_\forall F$ iff $P' \models F$ for every maximal consistent subset $P' \subseteq P$.

Example 3.3 Suppose P is the GHP below:
1 : $p \leftarrow q$
2 : $\neg p \leftarrow q$
3 : $q \leftarrow$
Clearly P is inconsistent. P has three maximal consistent subsets, viz. $P_1 = \{1,2\}, P_2 = \{2,3\}, P_3 = \{1,3\}$.
$SS(P_1) = \{\neg q\}$,
$SS(P_2) = \{q, \neg p\}$
$SS(P_3) = \{p, q\}$.

[1] We are grateful to Professor Newton da Costa for suggesting that $\vdash_\exists$ entailment may be a useful concept. The basic intuition behind $\vdash_\forall$ entailment is not entirely new. The idea of using maximal consistent subsets for hypothetical reasoning goes back to Rescher [12] whose work was later adapted to artificial intelligence by Ginsberg [5]. However, the technical properties of $\vdash_\exists$ and $\vdash_\forall$ entailment have not been studied carefully thus far and this is one of the things we do in this paper.

Thus, $P \vdash_{\exists} p$, $P \vdash_{\exists} \neg p$, $P \vdash_{\exists} q$ and $P \vdash_{\exists} \neg q$. There is no ground literal L such that $P \vdash_{\forall} L$.

Let us try to get some intuition. In $\vdash_{\exists}$ entailment, we adopt an *optimistic* approach. If P is inconsistent, then we say that L is true iff L is a consequence of *some* consistent subset of P. However, $\vdash_{\forall}$ is more *cautious*. It is not easily willing to admit that anything is true. For us to conclude L using $\vdash_{\forall}$ entailment, we must, intuitively, find all possible causes of inconsistency. If, after eliminating the cause of the inconsistency in all possible ways, it turns out that L is true in each scenario, only then do we consider L to be true. Intermediate concepts of inconsistency may also be devised such as the one in [9] where the concept of a "recoverable" literal is used.

Example 3.4 Consider the murder example of Section 2. Intuitively, a formula is $\vdash_{\forall}$ entailed iff it is true in all possible consistent scenarios. Thus, for example, the fact that Bill was alive when dinner was finished is $\vdash_{\forall}$ entailed by the evidence because it is true irrespective of whose version of the evidence we choose to believe. Likewise, the fact that Tom could not have been the murderer is clearly $\vdash_{\forall}$ entailed by the evidence.
On the other hand, for each person (except Tom) who had dinner with Bill that night, there is a scenario in which he could be the murderer. Thus, $\vdash_{\exists}$ entailment allows us to conclude, for example, that Carl is the murderer.
In effect, we can *use* $\vdash_{\exists}$ entailment in order to identify suspects, rather than to identify the murderer. $\vdash_{\exists}$ entailment tells us who we may safely ignore as a candidate murderer.

There is one feature of Example 3.3 that some readers may find curious. This concerns $\vdash_{\exists}$: Here, $P \vdash_{\exists} p$ and $P \vdash_{\exists} \neg p$, but $P \not\vdash_{\exists} (p \,\&\, \neg p)$. A brief discussion of this is in order. Even though p is true in some maximal consistent subset of P and likewise $\neg p$ is true in some maximal consistent subset of P, these two maximal subsets are different. In fact, there cannot be a *single* consistent subset of P in which *both* p and $\neg p$ are true. So even though P exhibits this kind of classical inconsistency with respect of $\vdash_{\exists}$ entailment, this inconsistency is not trivializing, i.e. the existence of such an inconsistency does not cause all formulas in our language to become $\vdash_{\exists}$ entailed by P.
Formally, some of these properties may be stated below:

Proposition 3.1 *Suppose P is a GHP. If P is consistent, then the following sentences are equivalent for all ground literals L:*

1. $P \models L$.
2. $P \vdash_{\exists} L$.
3. $P \vdash_{\forall} L$. □

Proposition 3.2 *Suppose P is a GHP (possibly inconsistent) and L, L_1, L_2 are ground literals. Then:*

1. *if $P \vdash_{\exists} (L_1 \,\&\, L_2)$, then $P \vdash_{\exists} L_1$ and $P \vdash_{\exists} L_2$.*
2. *In general, $P \vdash_{\exists} L_1$ and $P \vdash_{\exists} L_2$ do not imply that $P \vdash_{\exists} (L_1 \,\&\, L_2)$.*
3. *$P \vdash_{\exists} F$ for all tautologies F of classical logic.*

Proof. (1) Suppose $P \vdash_\exists (L_1 \& L_2)$. Then there is a maximal consistent subset $Q \subseteq P$ such that $Q \models L_1 \& L_2$. Hence, $Q \models L_1$. Thus, $P \vdash_\exists L_1$. Similarly for L_2.
(2) Immediate from Example 3.3.
(3) Suppose F is a tautology of classical logic. Then F is a logical consequence of the empty set, and hence F is a logical consequence of each consistent subset of P.
$\square$
The above proposition shows that the tautologies of classical logic hold with respect to $\vdash_\exists$- entailment. Similar properties hold for $\vdash_\forall$-entailment.

Theorem 2 *Suppose P is a GHP, and L, L_1, L_2 are ground literals. Then:*

1. *there is no ground literal L such that $P \vdash_\forall L$ and $P \vdash_\forall \neg L$.*
2. *$P \vdash_\forall (L_1 \& L_2)$ iff $P \vdash_\forall L_1$ and $P \vdash_\forall L_2$.*
3. *$P \vdash_\forall F$ for all tautologies F of classical logic.*

Proof. (1) Suppose $P \vdash_\forall L$ and $P \vdash_\forall \neg L$. Hence, for each maximal consistent subset Q of P, $Q \models L$ and $Q \models \neg L$, which contradicts our assumption that Q is maximal consistent. This means that there is no maximal consistent subset of P, which is impossible by Theorem 1.
(2) Suppose $P \vdash_\forall (L_1 \& L_2)$. Then $L_1 \& L_2$ is a logical consequence of every maximal consistent subset Q of P. Hence, each maximal consistent subset Q of P has L_1 and L_2 as a logical consequence, i.e. $Q \vdash_\forall L_1$ and $Q \vdash_\forall L_2$.
Suppose $P \vdash_\forall L_1$ and $P \vdash_\forall L_2$. Then L_1 and L_2 are both true in every maximal consistent subset Q of P, i.e. $P \vdash_\forall (L_1 \& L_2)$.
(3) The proof proceeds along the same lines as the proof of Proposition 3.2(3). $\square$

Example 3.5 Let P be:

$$p \leftarrow q$$
$$\neg p \leftarrow q$$
$$r \leftarrow$$
$$q \leftarrow$$

In this case, $P \vdash_\forall r$. But $P \not\vdash_\forall q$ and $P \not\vdash_\forall p$ and $P \not\vdash_\forall \neg q$ and $P \not\vdash_\forall \neg p$.

Thus, unlike $\vdash_\exists$ which can cause both L and $\neg L$ (but never $(L \& \neg L)$) to be inferred from a program, $\vdash_\forall$ does not allow this. However, $\vdash_\forall$ allows very few conclusions to be drawn. The following result is an immediate consequence of the fact that $CONS(P)$ is always non-empty.

Proposition 3.3 *Suppose P is any GHP and L any ground literal. If $P \vdash_\forall L$, then $P \vdash_\exists L$.* $\square$

We now demonstrate $\vdash_\exists$ and $\vdash_\forall$ entailment on a simple example.

Example 3.6 Consider the program P below:
1: $b \leftarrow a$
2: $\neg b \leftarrow a$
3: $a \leftarrow c$
4: $a \leftarrow$
5: $c \leftarrow$
There are four maximal consistent sets: $P_1 = \{1,2,3\}, P_2 = \{1,2,5\}, P_3 = \{1,3,4,5\}$ and $P_4 = \{2,3,4,5\}$. $SS(P_1) = \{\neg a, \neg c\}$, $SS(P_2) = \{\neg a, c\}$, $SS(P_3) = \{a, b, c\}$, $SS(P_4) = \{a, \neg b, c\}$. The literals that are $\vdash_\exists$-entailed by P are: $\{\neg a, \neg c, c, a, b, \neg b\}$. The set of literals $\vdash_\forall$-entailed by P is $\emptyset$.

In simple examples, such as Examples 3.3 and 3.6, for any two distinct maximal consistent subsets P_1, and P_2, there is usually a literal ℓ such that $\ell \in SS(P_1)$ and $\neg\ell \in SS(P_2)$. The following more complex example shows that this need not always be the case.

Example 3.7 Consider the set of clauses:

1:	$\neg p \leftarrow$
2:	$\neg q \leftarrow$
3:	$p_3 \leftarrow$
4:	$q_3 \leftarrow$
5:	$q_1 \leftarrow \neg p_1 \,\&\, \neg p_2$
6:	$q_1 \leftarrow p_1 \,\&\, \neg p_2$
7:	$q_1 \leftarrow \neg p_1 \,\&\, p_2$
8:	$q_2 \leftarrow p_1 \,\&\, \neg p_2$
9:	$q_2 \leftarrow \neg p_1 \,\&\, p_2$
10:	$q_2 \leftarrow \neg p_1 \,\&\, \neg p_2$
11:	$p_1 \leftarrow \neg q_1 \,\&\, \neg q_2$
12:	$p_1 \leftarrow q_1 \,\&\, \neg q_2$
13:	$p_1 \leftarrow \neg q_1 \,\&\, q_2$
14:	$p_2 \leftarrow q_1 \,\&\, \neg q_2$
15:	$p_2 \leftarrow \neg q_1 \,\&\, q_2$
16:	$p_2 \leftarrow \neg q_1 \,\&\, \neg q_2$
17:	$p \leftarrow p_1 \,\&\, p_2 \,\&\, p_3$
18:	$q \leftarrow q_1 \,\&\, q_2 \,\&\, q_3$

There are several maximal consistent sets here. Let $P_1 = \{1, \ldots, 16, 17\}$ and $P_2 = \{1, \ldots, 16, 18\}$. P_1 and P_2 are maximal consistent subsets of P. $SS(P_1) = \{\neg p, \neg q, p_3, q_3, q_1, q_2\}$, $SS(P_2) = \{\neg p, \neg q, p_3, q_3, p_1, p_2\}$. Note that there is no literal in $SS(P_1)$ whose negation is in $SS(P_2)$.

Before concluding this section, we briefly observe that the problem "Given as inputs, a GHP P, and a literal L, determining whether $P \vdash_{\forall} L$" is Π_2^p-complete and the analogous problem "Given as inputs, a GHP P, and a literal L, determining whether $P \vdash_{\exists} L$" is Σ_2^p-complete. The former is true because
$P \vdash_{\forall} L \Leftrightarrow$
$([(\forall Q \subseteq P)(Q \text{ is consistent} \,\&\, (\forall Q^*)(Q \subset Q^* \subseteq P \rightarrow Q^* \text{ is inconsistent})] \rightarrow Q \models L)$

This is a Π_2^p problem because it involves a universal quantification over an NP-complete problem (viz. checking the consistency of Q, and making a polynomial-number of inconsistency checks of Q^*). The Σ_2^p-result for optimistic entailment follows analogously, together with the observation that it involves an existential quantification over the same NP-complete problem.

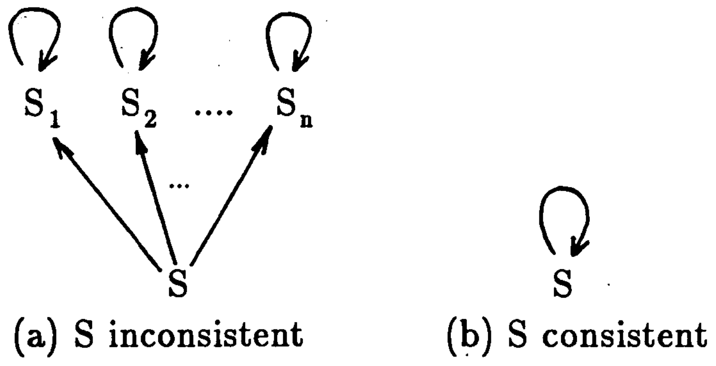

Figure 1: Graphical Representation of $PK(S)$

4 Kripke Semantics and a Fixpoint Operator

In this section, we develop a Kripke-style model theory for optimistic and cautious entailment. We also develop a fixpoint operator for the optimistic semantics. We assume that a GHP is a finite set of ground clauses. Given a GHP P, we use $\mathbf{D}(P)$ to denote the set of all ground disjunctions of literals (including the empty disjunction) expressible using the language of P.

Definition 4.1 An elementary structure of the language of P (*e-structure*, for short) is any subset of $\mathbf{D}(P)$.

Definition 4.2 An e-structure S of a GHP P is said to be a consistent structure (*c-structure*, for short) iff S has a model in the sense of classical logic.

Definition 4.3 Suppose P is a GHP and S is an e-structure of P. The *paraconsistent Kripke structure* (PK-structure, for short) *based on* S is a pair $(Int(S), Edge(S))$ defined as follows:

1. If S is a c-structure, then $Int(S) = \{S\}$ and $Edge(S) = \{(S, S)\}$.
2. If S is not a c-structure, then:
   (a) $Int(S) = \{S\} \cup \{S' \mid S' \subseteq S$ and S' is a c-structure and there is no $J \subseteq S$ such that J is a c-structure and $S' \subset J\}$.
   (b) $Edge(S) = \{(S, J) \mid J \in (Int(S) - \{S\})\} \cup \{(J, J) \mid J \in (Int(S) - \{S\})$.

Figure 1 shows a graphical representation of $PK(S)$. In Figure 1(a), $S_1, \ldots, S_n$ are maximal c-substructures of S.

Example 4.1 Suppose P is a GHP written in the language consisting of three propositional symbols p, q and r, and S is the e-structure $\{p, q, \neg q\}$, then $PK(S)$ is the pair $(Int(S), Edge(S))$ where:
$Int(S) = \{S, \{p, q\}, \{p, \neg q\}\}$
$Edge(S) =$ (the reflexive closure of $\{(S, \{p, q\}), (S, \{p, \neg q\})\}$) $- \{(S, S)\}$.

Suppose $\mathcal{L}$ is a first order language. We extend $\mathcal{L}$ to a modal language, denoted $\mathcal{L}_{\mathcal{M}}$ defined as follows:

1. Every wff of classical logic is a wff of $\mathcal{L}_{\mathcal{M}}$.
2. If F is a wff of $\mathcal{L}_{\mathcal{M}}$, then $\Diamond F$ and $\Box F$ are wffs of $\mathcal{L}_{\mathcal{M}}$. (Intuitively, $\Diamond F$ is to be read as "F is possible", while $\Box F$ is to be read as "F is necessary".)
3. If F and G are wffs of $\mathcal{L}_{\mathcal{M}}$, then $F \,\&\, G$, $F \vee G$, $\neg F$, $F \leftarrow G$, $F \leftrightarrow G$ and $(\forall x)F$ and $(\exists x)F$ are wffs of $\mathcal{L}_{\mathcal{M}}$.

$\mathcal{L}_{\mathcal{M}}$ is interpreted by a PK-structure based on an e-structure S defined as follows:

Definition 4.4 Suppose S is an e-structure, and let $E = (Int(S), Edge(S))$ be the PK-structure based on S. Let $w \in Int(S)$. Then we say that E, w *satisfies* F, denoted $E, w \mapsto F$ as follows:

1. If F is a wff of classical logic, then:
   (a) (F an atom) $E, w \mapsto F$ iff F is a logical consequence of w
   (b) ($F = \neg G$) $E, w \mapsto F$ iff G is not a logical consequence of w (here, we assume G is an atom)
   (c) ($F = G \,\&\, H$) $E, w \mapsto F$ iff $E, w \mapsto G$ and $E, w \mapsto H$
   (d) ($F = G \vee H$) $E, w \mapsto F$ iff $E, w \mapsto G$ or $E, w \mapsto H$
   (e) ($F = G \rightarrow H$) $E, w \mapsto F$ iff $E, w \mapsto H$ or $E, w \not\mapsto G$
   (f) Satisfaction of formulas whose leading connective is a quantifier is defined in the usual way.
2. Suppose $F \equiv \Diamond G$. Then $E, w \mapsto F$ iff for some w' such that $(w, w') \in Edge(S)$, $E, w' \mapsto G$.
3. Suppose $F \equiv \Box G$. Then $E, w \mapsto F$ iff for each w' such that $(w, w') \in Edge(S)$, $E, w' \mapsto G$.
4. Satisfaction of formulas whose leading connectives are conjuncts, disjuncts, implications, iff, and the quantifiers are defined in the usual way.

Using the notion of a PK-structure based on an e-structure S, we may define the model theoretic semantics of the logics corresponding to $\vdash_{\exists}$ and $\vdash_{\forall}$ entailment.

Definition 4.5 Suppose F is a formula of classical logic and S is an e-structure. Let E be the PK-structure determined by S. We say that

1. $S \models_{\exists} F$ iff $E, S \mapsto \Diamond F$.
2. $S \models_{\forall} F$ iff $E, S \mapsto \Box F$.

Suppose Δ is a set of formulas. $S \models_{\exists} \Delta$ iff $S \models_{\exists} \delta$ for all $\delta \in \Delta$.

In order to show the equivalence of $\vdash_{\exists}$ and $\models_{\exists}$, we need a definition.

Definition 4.6 Given a clause $C \equiv$

$$D \leftarrow L_1 \,\&\, \ldots \,\&\, L_n$$

the disjunctive form of C, denoted $disj(C)$, is the clause:

$$D \vee \neg L_1 \vee \cdots \vee \neg L_n.$$

The *disjunctive form*, $disj(P)$, of a GHP P is the set $\{disj(C) \mid C \in P\}$.

Proposition 4.1 Suppose P is a GHP and D is a ground disjunction. Then:

1. $P \vdash_\exists D$ iff $P \models_\exists D$
2. $P \vdash_\forall D$ iff $P \models_\forall D$.

Proof. We prove (1) above. the proof of (2) is similar.
Suppose $P \vdash_\exists D$. Then $disj(P) \vdash_\exists D$. Hence, there is a consistent subset $P' \subseteq disj(P)$ such that $P' \models D$. Suppose now that I is an e-structure such that $I \models_\exists P$. Clearly, $disj(P) \subseteq I$ and hence, $P' \subseteq I$. Extend P' to a maximal consistent subset of I. This maximal consistent subset of I must make D true.
Conversely, suppose $P \models_\exists D$. Consider the e-structure $disj(P)$. As $disj(P) \models_\exists P$, there is a maximal consistent subset $I^* \subseteq disj(P)$ such that D is true in I^*. Let $P' = I^*$. This completes the proof. □
Given a GHP P, we observe that P may entail a ground literal even though there is no clause in P having an instance containing that ground literal as the head. To see this observe that the program P below entails $\neg b$:
$a \leftarrow b$
$\neg a \leftarrow b$
There is no clause in P with $\neg b$ as the head. Now add the contrapositives to P.
$\neg b \leftarrow \neg a$ (contrapositive of first clause)
$\neg b \leftarrow a$ (contrapositive of second clause)
The expanded program is equivalent to the original program P. The addition of contrapositives now yields a clause with $\neg b$ in the head.
Consider now the program Q below:

$$p \leftarrow a$$

$$p \leftarrow b$$

$$a \leftarrow \neg b$$

$$b \leftarrow \neg a$$

We would like to define a fixed-point operator which yields p as a consequence of Q. Moreover, $(a \vee b)$ should also be a consequence of Q.
Based on the optimistic notion of entailment, we now develop a fixed point semantics for $\vdash_\exists$-entailment. We start by observing that given a clause C, there may be disjunctions, D, of literals that are logically entailed by the program P, but do not appear in the head of C. The implicational form of clause C, defined below, rewrites C in all possible disjunctive ways.

Definition 4.7 Suppose $C \equiv$

$$L \leftarrow L_1 \,\&\, \ldots \,\&\, L_n$$

is a clause. The *implicational form*, $IF(C)$, of C is the set of clauses $\{L'_1 \vee \cdots \vee L'_m \leftarrow Body \mid \{L'_1, \ldots, L'_m\} \cup \{\neg K \mid K \in Body\} = \{L, \neg L_1, \ldots, \neg L_n\}$ and $m > 0\}$. The *normal form*, $NF(P)$ of a generally Horn program P is then defined to be:

$$NF(P) = \bigcup_{C \in P} IF(C).$$

Note that $disj(C) \in IF(C)$ and $disj(P) \subseteq NF(P)$. Given a ground disjunction D and a GHP P, we use the notation $\mathbf{sub}(D, P)$ to denote the set $\{C \mid C$ is a clause in $NF(P)$ such that the head of C subsumes $D\}$. Thus, if D is not subsumed by the head of any clause in $NF(P)$, then $\mathbf{sub}(D, P) = \emptyset$.

Definition 4.8 Suppose P is a GHP and S is an e-structure. We define an operator that maps e-structures to e-structures. Let $TAUT$ denote the set of all tautologous clauses expressible in our language.
$V_P(S) = TAUT \cup \{D \mid \mathbf{sub}(D, P) \neq \emptyset$ and such that:

1. for all $1 \leq i \leq n$, there exists a disjunction E_i (possibly empty) of ground literals such that $PK(S), S \mapsto \Diamond(\bigvee_{i=1}^{k}(\bigwedge_{j=1}^{n_i}(L_j^i \vee E_i)))$ where $\mathbf{sub}(D, P) = \{C_1, \ldots, C_k\}$ for $k \geq 1$ and each C_i is of the form

$$C_i \equiv \; D'_i \leftarrow L_1^i \,\&\, \cdots \,\&\, L_{n_i}^i$$

and

2. for all $1 \leq i \leq n$, the smallest factor of $(D'_i \vee E_i)$ subsumes D.

Remark 4.1 When a GHP is a disjunctive logic program in the sense of Minker and Rajasekar [11], (i.e. clause heads and clause bodies may contain no negated atoms), our operator is essentially the same as that of Rajasekar and Minker. The only difference is that in our case, the presence of subsumed clauses is explicit in $V_P(S)$, while in the case of Rajasekar and Minker, it is implicit.

To see how V_P works, consider the following example.

Example 4.2 Suppose P consists of the following five clauses:

1: $p \leftarrow q \,\&\, \neg q$
2: $r \leftarrow p$
3: $r \leftarrow \neg p$
4: $q \leftarrow$
5: $\neg q \leftarrow$

(Note here that $NF(P)$ contains more clauses, but these are not needed for this example.) Let S be the e-structure $\{q, \neg q\}$. Then the set of ground atoms in $V_P(S)$ is the set $\{q, \neg q, r\}$. Let us explain two things: (1) why $r \in V_P(S)$ and (2) why $p \notin V_P(S)$.
(1) Note that $\mathbf{sub}(r, P) = \{2, 3\}$. In particular, using the notation of Definition 4.8, we may assume the E_i's to be the empty clause. Observe that

$$PK(S), S \mapsto \Diamond(p \vee \neg p).$$

To see this note that in this case, $Int(S) = \{S, \{q\}, \{\neg q\}\}$. It is easy to see that $p \vee \neg p$ is true in both c-structures $\{q\}, \{\neg q\}$ that are accessible from world S.
(2) To see why p cannot be in $V_P(S)$, observe that the only clause with p is the head is clause (1). The antecedent of clause (1) is a flat contradiction which cannot be true in either $\{q\}$ or $\{\neg q\}$.

Intuitively, the operator V_P is supposed to capture the notion of $\models_\exists$ entailment.

Example 4.3 Consider the consistent GHP P below:

$$p \leftarrow a$$

$$\neg p \leftarrow a$$

$$b \leftarrow \neg a$$

$\neg a$ is a logical consequence of P, and hence b should be a logical consequence of P. Here, $NF(P)$ is the program:

1. $p \leftarrow a$	4. $\neg a \leftarrow \neg p$	7. $p \vee \neg a \leftarrow$
2. $\neg p \leftarrow a$	5. $\neg a \leftarrow p$	8. $\neg p \vee \neg a \leftarrow$
3. $b \leftarrow \neg a$	6. $a \leftarrow \neg b$	9. $b \vee a \leftarrow$

The least fixed-point of V_P is constructed as follows:
$V_P \uparrow 0 = \emptyset$
$V_P \uparrow 1$ contains $\neg a$, $b \vee a$ together with tautologies and subsumed clauses
$V_P \uparrow 2$ contains b, $V_P \uparrow 1$, together with tautologies and subsumed clauses

We end this section by proving the soundness and completeness of the computation captured by the fixed-point operator V_P.

Theorem 3 Suppose P is a GHP and D is any ground disjunction. Then $D \in V_P \uparrow \omega$ iff $P \vdash_\exists D$.

Proof. We first show that if $D \in V_P \uparrow \omega$ then $P \vdash_\exists D$.
Suppose $D \in V_P \uparrow \omega$. Then there is an integer $n < \omega$ such that $D \in V_P \uparrow n$. We proceed by induction on n.
Base Case. ($n = 0$) Trivial.
Inductive Case. ($n = r + 1$) Suppose $\mathrm{sub}(D, P) = \{C_1, \ldots, C_k\}$ where

$$C_i \equiv D_i' \leftarrow L_1^i \,\&\cdots\&\, L_{n_i}^i.$$

Then, as $D \in V_P \uparrow (r+1)$, it follows that

$$PK(V_P \uparrow r), V_P \uparrow r \mapsto \Diamond(\bigvee_{i=1}^{k} (\bigwedge_{j=1}^{n_i} (L_i \vee E_i)))$$

where the E_i's are ground clauses (possibly empty). Let $M_1, \ldots, M_s$ be the maximal c-structures that are subsets of $V_P \uparrow r$. From the above, we know that there is a $1 \le j \le s$ such that $M_j \mapsto (\bigvee_{i=1}^{k}(\bigwedge_{j=1}^{n_i}(L_i \vee E_i)))$, and hence it follows by the induction hypothesis that there is a maximal consistent subset P_j of P such that

$P_j \models (\bigvee_{i=1}^{k}(\bigwedge_{j=1}^{n_i}(L_i \vee E_i)))$. As P_j is maximal and consistent, it must also entail D. Therefore, $P \vdash_{\exists} D$.
To prove the converse, i.e. to show that if $P \vdash_{\exists} D$, then $D \in V_P \uparrow \omega$, we proceed as follows: As $P \vdash_{\exists} D$, there is a maximal consistent subset Q of P such that $Q \models D$. This is classical logic entailment. Transform Q into a disjunctive logic program (in the sense of Rajasekar and Minker [11]) as follows: if

$$A_1 \vee \cdots \vee A_n \vee \neg B_1 \vee \cdots \neg B_m \leftarrow D_1 \,\&\ldots\&\, D_r \,\&\, \neg E_1 \,\&\cdots\&\, \neg E_s$$

is in D, then replace it by the disjunctive clause:

$$A_1 \vee \cdots \vee A_n \vee E_1 \vee \cdots \vee E_s \leftarrow B_1 \,\&\cdots\&\, B_m \,\&\, D_1 \,\&\ldots\&\, D_r$$

The resulting program, called Q', is a disjunctive logic program in the sense of Rajasekar and Minker and hence it has the same logical consequences as D. As the V_P operator of ours is equivalent to that of Rajasekar's and Minker's for disjunctive logic programs, it follows by a result of theirs that $D \in lfp(V_P)$ and hence $D \in V_P \uparrow \omega$.
This completes the proof. □

5 Stratification and Stability

So far, we have assumed that negation (the symbol $\neg$) represents a "classical" form of negation, i.e. in order to conclude $\neg A$ for some ground atom A, one must explicitly establish the truth of $\neg A$ rather than reason from the lack of a proof of A. However, it is now widely accepted that requiring the explicit specification of negative information causes knowledge bases often to grow very large. However, as argued by Gelfond and Lifschitz [7] and Kowalski and Sadri [10], in many cases both classical and non-monotonic modes of negation are required. In this section, we extend the optimistic and cautious semantics to incorporate non-monotonic negation. Then we show how the concepts of stratification and stability can be extended to this framework.

Definition 5.1 If $L, L_1, \ldots, L_n, L'_1, \ldots, L'_m$ are literals, then

$$L \leftarrow L_1 \,\&\ldots\&\, L_n \,\&\mathbf{not}\, L'_1 \,\&\ldots\&\, \mathbf{not}\, L'_m$$

is called an *extended program clause*. An *extended GHP* (called EGHP, for short), is a finite set of extended program clauses.

Here, the symbol **not** denotes a non-monotonic mode of negation. As usual, we deal with the set of all ground instances of clauses in an extended GHP. Now, we extend the standard definitions of stability to deal with non-monotonic negation using the optimistic and cautious semantics.

Definition 5.2 Suppose P is an EGHP and X is a set of ground literals. The *transformation* of P w.r.t. X is the logic program $G(P, X)$ obtained as follows:

1. if C is a program clause in P of the form

$$L \leftarrow L_1 \,\&\ldots\&\, L_n \,\&\, \mathbf{not}\, H_1 \,\&\ldots\&\, \mathbf{not}\, H_m$$

such that for all $1 \leq i \leq m$, $H_i \notin X$, then

$$L \leftarrow L_1 \,\&\ldots\&\, L_n$$

is in $G(P, X)$.

2. Nothing else is in $G(P, X)$.

Definition 5.3 Given an EGHP P, we define two operators that map sets of ground literals to sets of ground literals as follows:
$\mathbf{A}_P(X) = \{\psi \mid \psi$ is a ground literal such that $G(P, X) \vdash_\forall \psi\}$.
$\mathbf{E}_P(X) = \{\psi \mid \psi$ is a ground literal such that $G(P, X) \vdash_\exists \psi\}$.

Definition 5.4 A set X of ground literals is

1. an **A**-answer set for EGHP P iff $\mathbf{A}_P(X) = X$
2. an **E**-answer set for EGHP P iff $\mathbf{E}_P(X) = X$

In general, EGHPs may have zero, one or many answer sets. The notion of an answer set is similar to the notion of a stable model; however, an answer set need not be a model of P.

Example 5.1 Consider the program:

$$a \leftarrow \mathbf{not}\, a \qquad (1)$$
$$\neg a \leftarrow \mathbf{not}\, a \qquad (2)$$

This program has an **A**-answer set $\emptyset$, but no **E**-answer set.

Example 5.2 Consider the program:

$$a \leftarrow \mathbf{not}\, b \qquad (3)$$
$$\neg a \leftarrow \mathbf{not}\, b \qquad (4)$$
$$b \leftarrow a \,\&\, \neg a \qquad (5)$$

This program has an **A**-answer set $\emptyset$ and an **E**-answer set $\{a, \neg a\}$.

Example 5.3 Consider the program:

$$a \leftarrow \mathbf{not}\, a \,\&\, \mathbf{not}\, b \qquad (6)$$
$$b \leftarrow \qquad (7)$$
$$\neg b \leftarrow \qquad (8)$$

This program has an **E**-answer set $\{b, \neg b\}$, but no **A**-answer set.

We may wonder under what conditions an EGHP has a unique **E**-answer set or **A**-answer set. We now study this problem and provide a sufficient, but not necessary condition to guarantee the existence of such answer sets. This is achieved by extending the concept of stratification to EGHPs.
For logic programs, stratification may be defined in terms of a level mapping of ground atoms. In our case, a level mapping is a function from the set of ground literals to the set of non-negative integers. The value of a literal L under level mapping ℓ is written as $\ell(L)$. The levels of a program are assumed to range from 0 to k for some integer k. The clauses of the program are placed in strata S_i, $0 \le i \le k$, by placing a clause whose head has level i into S_i. For the definitions below, we use the generic clause

$$L \leftarrow L_1 \,\&\ldots\&\, L_n \,\&\, \mathbf{not}\, L'_1 \,\&\ldots\&\, \mathbf{not}\, L'_m.$$

We start with an (intermediate) definition.

Definition 5.5 ([8]) An extended GHP P is said to be *OD-stratified* iff there is a level mapping ℓ such that for every clause $C \in grd(P)$ of the above form, $\ell(L_i) \leq \ell(L)$ and $\ell(L'_j) < \ell(L)$ for all $1 \leq i \leq n$ and $1 \leq j \leq m$.

Basically, OD-stratification treats all literals equally and does not allow recursion through non-monotonic negation (**not**), but allows recursion through classical negation ($\neg$).

Definition 5.6 The *switched form* $SF(C)$ of a (generic) clause C is the set of clauses $\{\neg L_i \leftarrow \neg L \;\&\; L_1 \;\&\ldots\&\; L_{i-1} \;\&\; L_{i+1} \;\&\ldots\&\; L_n \;\&\; \mathbf{not}\, L'_1 \;\&\ldots\&\; \mathbf{not}\, L'_m\}$ obtained from C by switching (and negating) the literal in the head of C with a literal in the body not preceded by **not**. The *switched form* $SF(P)$, of an EGHP P is defined as $SF(P) = \bigcup_{C\in P} SF(C)$.

Definition 5.7 An EGHP P is called **E**-*stratified* iff $SF(P)$ is OD-stratified.

Definition 5.8 An EGHP P is called **A**-*stratified* iff P is **E**-stratified and for every ground atom A, $\ell(A) = \ell(\neg A)$.

Clearly, every **A**-stratified EGHP is also **E**-stratified; the latter also implies that the EGHP is OD-stratified. However, OD-stratification does not necessarily imply **E**-stratification, and **E**-stratification does not necessarily imply **A**-stratification.

Example 5.4 Let P be:

$$b \leftarrow a \;\&\; \mathbf{not}\, \neg a.$$

P is OD-stratified by $\ell(\neg a) = \ell(\neg b) = \ell(a) = 0$ and $\ell(b) = 1$. $SF(P)$ in this case is:

$$b \leftarrow a \;\&\; \mathbf{not}\, \neg a \tag{9}$$
$$\neg a \leftarrow \neg b \;\&\; \mathbf{not}\, \neg a \tag{10}$$

$SF(P)$ is not OD-stratified because that would require $\ell(\neg a) < \ell(\neg a)$. Hence, P is not **E**-stratified.

Example 5.5 Let P be:

$$\neg b \leftarrow a \;\&\; \mathbf{not}\, b.$$

Now $SF(P)$ is:

$$\neg b \leftarrow a \;\&\; \mathbf{not}\, b \tag{11}$$
$$\neg a \leftarrow b \;\&\; \mathbf{not}\, b \tag{12}$$

$SF(P)$ is OD-stratified by $\ell(a) = \ell(b) = 0$, $\ell(\neg a) = \ell(\neg b) = 1$. Hence, P is **E**-stratified. However, P is not **A**-stratified because any level mapping ℓ must have $\ell(b) < \ell(\neg b)$.

Now we show how stratification provides a sufficient condition for stability in our framework of non-monotonic negation within inconsistent knowledge bases.

Theorem 4 If P is a function-free **E**-stratified EGHP, then P has a unique **E**-answer set.

Proof. By the hypothesis, there is a level mapping ℓ for $SF(P)$ such that for every clause $C \in SF(P)$, $\ell(L_i) \leq \ell(L)$ and $\ell(L') < \ell(L)$ for all $1 \leq i \leq n$ and $1 \leq j \leq m$, where C is written in the standard form (cf. Definition 5.1). Let $S_0, \ldots, S_n$ be the strata generated by this mapping and for $0 \leq i \leq n$, let $T_i = \{L \mid \ell(L) = i\}$. We construct an **E**-answer set M as follows:

$$M_0 = \{L \in T_0 \mid S_0 \vdash_\exists L\};$$
$$M_{i+1} = \{L \in T_{i+1} \mid G(S_i, \textstyle\bigcup_{j=0}^{i} M_j) \vdash_\exists L\} \text{ for } 1 \leq i \leq n;$$
$$M = \textstyle\bigcup_{i=0}^{n} M_i.$$

We need to show that M is an **E**-answer set. We start by observing that for every literal L and set of literals V, $G(P,V) \vdash_\exists L$ iff $SF(G(P,V)) \vdash_\exists L$. This is so because every clause in $SF(G(P,V))$ is logically equivalent to some clause in $G(P,V)$. Now, note that $SF(G(P,V)) = G(SF(P),V)$ because the non-monotonically negated literals are not modified by SF. Hence, $G(P,V) \vdash_\exists L$ iff $G(SF(P,V)) \vdash_\exists L$. To show that M is an **E**-answer set, we must obtain $M = \{L \mid G(P,M) \vdash_\exists L\}$, or by the previous discussion, $M = \{L \mid G(SF(P),M) \vdash_\exists L\}$. But the **E**-stratification of P implies that

$$G(SF(P),M) = \bigcup_{i=0}^{n} G(S_i, \bigcup_{j=0}^{i-1} M_j)$$

where $\bigcup_{j=0}^{i-1} M_j = \emptyset$, because for every clause in strata i, the non-monotonically negated clauses cannot be added to M at any level greater than or equal to i. The result follows from the construction of M and the fact that if $G(SF(P),M) \vdash_\exists L$, then there must be a clause in $SF(P)$ with L as the head.

We still need to show that M is the unique **E**-answer set for P. Suppose M' is any **E**-answer set for P. We show that $M = M'$ by showing that $M_i = M'_i$ for all $1 \leq i \leq n$ where $M'_i = M' \cap \{L \mid \ell(L) = i\}$.
Base Case. ($i = 0$) In this case, for every clause in $SF(P)$ in stratum S_0, there are no occurrences of **not**. Hence, $G(S_0, M_0) = G(S_0, M')$. Thus, $\mathbf{E}_P(M_0) = \mathbf{E}_P(M'_0)$. As M_0 and M'_0 must be **E**-answer sets for S_0, $M_0 = \mathbf{E}_P(M_0) = \mathbf{E}_P(M'_0) = M'_0$.
Inductive case. ($i > 0$) Assume that $M_j = M'_j$ for all $j < i$. By the **E**-stratification of P, $G(S_i, M_i) = G(S_i, M'_i)$ and then by reasoning similar to the base case, $M_i = \mathbf{E}_P(M_i) = \mathbf{E}_P(M'_i) = M'_i$. □
E-stratification is a sufficient, but not a necessary condition for an EGHP to have a unique **E**-answer set. In particular, the program of Example 5.3 is not **E**-stratified, but it has $\{b, \neg b\}$ as its unique **E**-answer set. The next example shows that **E**-stratification is not a sufficient condition for an EGHP to have an **A**-answer set.

Example 5.6 Consider the program:

$$a \leftarrow \mathbf{not}\, b \quad (13)$$
$$b \leftarrow \quad (14)$$
$$\neg b \leftarrow \mathbf{not}\, a \quad (15)$$

This program is **E**-stratified with $\ell(b) = \ell(\neg a) = 0$, $\ell(a) = 1$, $\ell(\neg b) = 2$. Here, $P = SF(P)$. However, there is no **A**-answer set. Note that this program is not **A**-stratified.

Theorem 5 If P is a function-free **A**-stratified EGHP, then P has a unique **A**-answer set.

Proof. The construction of the **A**-answer set is similar to the construction in Theorem 4 except for the substitution of $\vdash_{\forall}$ instead of $\vdash_{\exists}$. The key point in showing that M is an **A**-answer set is that for every ground atom A, since $\ell(A) = \ell(\neg A) = i$ for some level i, at that level either A is placed into M_i or $\neg A$ is placed into M_i or neither A nor $\neg A$ is placed into M_i. By the definition of **A**-stratification, it is impossible to add A at some level and $\neg A$ at another level. □

Example 5.1 shows that **A**-stratification is not a necessary condition for the existence of an **A**-answer set. The program of Example 5.1 is not **A**-stratified, but it has $\emptyset$ as its **A**-answer set.

6 Summary and Discussion

We have developed two semantics for inconsistent knowledge bases. Both semantics are based on the maximal consistent subsets of the inconsistent knowledge base. The cautious semantics accepts those statements which are true in all maximal consistent subsets, while the optimistic semantics accepts those statements which are true in at least one maximal consistent subset. We study various properties of these semantics and develop a Kripke-style model theory for the optimistic semantics. Finally, we extend our approach to include non-monotonic negation. Within this framework, we extend the concepts of stratification and stability from logic programming and show that stratification provides a sufficient condition for stability.

Reasoning with inconsistency in logic programs was first studied by Blair and Subrahmanian [2] whose work was subsequently expanded by Kifer and Lozinskii [9]. These works were grounded in multivalued logics. There are two significant differences between those approaches and that studied in this paper: first, when a database DB is consistent, the semantics of [2,9] may not always agree with the classical logic meaning of DB; both the optimistic and cautious approaches described here would agree with classical logic when DB is classically consistent. Second, the work described here includes support for non-monotonic negation via a stable model semantics. No support for non-monotonic negation was present in [2,9], though [9] discusses some ways non-monotonicity may occur. In particular, we present here, two kinds of stratification. Neither [2,9] did this.

A structure similar to maximal consistent subsets arises in the context of database updates [3,4]. Given a database DB and a new fact, f, to be inserted into the database, Fagin et. al. [3,4] define a *flock* to be the set of maximal consistent subsets of $DB \cup \{f\}$ that are supersets of $\{f\}$. In other words, priority is given to f over formulas in DB. This does not occur in our framework.

Finally, the work reported in this paper has been used as the formal theoretical basis for combining multiple knowledge bases [1].

Acknowledgements. This work has been supported by the Army Research Office under Grant Number DAAL-03-92-G-0225, by the Air Force Office of Scientific Research under Grant Number F49620-93-1-0065, and by the National Science Foundation under Grant Numbers IRI-9200898, IRI-9109755 and IRI-9357756. The work was also supported in part by ARPA/Rome Labs Contract F30602-93-C-0241 (ARPA Order Nr. A716).

References

[1] C. Baral, S. Kraus, J. Minker and V.S. Subrahmanian (1992) *Combining Knowledge Bases Consisting of First Order Theories*, Computational Intelligence, 8, 1, pps 45–71.

[2] H.A. Blair and V.S. Subrahmanian. (1989) *Paraconsistent Logic Programming*, Theoretical Computer Science, Vol. 68, pp 135–154. Preliminary version in: Lecture Notes in Computer Science, Vol. 287, Dec. 1987.

[3] R. Fagin, G. Kuper, J. Ullman, and M. Vardi. Updating Logical Databases. In *Advances in Computing Research*, volume 3, pages 1–18, 1986.

[4] R. Fagin, J.D. Ullman, and M.Y. Vardi. On the Semantics of Updates in Databases. In *ACM SIGACT/SIGMOD Symposium on Principles of Database Systems*, pages 352–365, 1983.

[5] M. Ginsberg. (1986) *Counterfactuals*, Artificial Intelligence.

[6] M. Gelfond and V. Lifschitz. (1988) *The Stable Model Semantics for Logic Programming*, in Proc. of the 5th Intl. Conf./Symp. on Logic Programming, pp 1070–1080, MIT Press.

[7] M. Gelfond and V. Lifschitz. (1990) *Logic Programs with Classical Negation*, in: Proc. of the 7th Intl. Conf. on Logic Programming, pp 579–597, MIT Press.

[8] J. Grant and V.S. Subrahmanian. (1995) Reasoning In Inconsistent Knowledge Bases, IEEE Trans. on Knowledge and Data Engineering, volume 7, pp 177-189.

[9] M. Kifer and E.L. Lozinskii. (1989) *RI: A Logic for Reasoning with Inconsistency*, 4-th Symposium on Logic in Computer Science, Asilomar, CA, pp. 253-262.

[10] R. Kowalski and F. Sadri. (1990) *Logic Programs with Exceptions*, in: Proc. 7th Intl. Conf. on Logic Programming, pp 598–613.

[11] J. Minker and A. Rajasekar. (1990) *A Fixpoint Semantics for Non-Horn Logic Programs*, J. of Logic Programming.

[12] N. Rescher. (1964) *Hypothetical Reasoning*, North-Holland.

Appendix

Formalization of the Murder Example

The various facts relating to the murder mystery are described below:

1. $present(al)$
2. $present(carl)$
3. $present(ed)$
4. $present(dick)$
5. $in\_hospital(tom)$
6. $\neg present(X) \leftarrow in\_hospital(X)$

7. $suspect(X) \leftarrow present(X)$
8. $\neg suspect(X) \leftarrow \neg present(X)$
9. $suspect(X) \leftarrow embezzler(X)$
10. $suspect(X) \leftarrow having\_affair(X)$
11. $embezzler(al)$
12. $having\_affair(carl)$
13. $in\_bathroom(carl)$
14. $in\_bathroom(al)$
15. $\neg in\_bathroom(Y) \leftarrow in\_bathroom(X)\ \&\ X \neq Y$
16. $carl \neq al$
17. $al \neq carl$
18. $in\_living\_room(dick)$
19. $in\_living\_room(ed)$
20. $\neg in\_kitchen(X) \leftarrow in\_bathroom(X)$
21. $\neg in\_kitchen(X) \leftarrow in\_living\_room(X)$
22. $murderer(al) \leftarrow \neg murderer(carl) \& \neg murderer(ed) \& \neg murderer(dick)$
23. $murderer(carl) \leftarrow \neg murderer(al) \& \neg murderer(ed) \& \neg murderer(dick)$
24. $murderer(ed) \leftarrow \neg murderer(carl) \& \neg murderer(al) \& \neg murderer(dick)$
25. $murderer(dick) \leftarrow \neg murderer(carl) \& \neg murderer(ed) \& \neg murderer(al)$
26. $suspect(X) \leftarrow murderer(X)$
27. $\neg murderer(X) \leftarrow murderer(Y)\ \&\ Y \neq X.$
28. $\neg murderer(X) \leftarrow in\_bathroom(X)$
29. $\neg murderer(X) \leftarrow in\_living\_room(X)$
30. AXIOMS SAYING that Carl, Al, Ed, and Tom are not equal.

Received June, 1994