RI: A Logic for Reasoning with Inconsistency

Michael Kifer*  
Department of Computer Science  
SUNY at Stony Brook  
Stony Brook, NY 11794, USA  
kifer@scs.sunysb.edu

Eliezer L. Lozinskii**  
Department of Computer Science  
The Hebrew University  
Jerusalem 91904, Israel  
lozinskii@humus.huji.ac.il

ABSTRACT

Most known computational approaches to reasoning have problems when facing inconsistency, so they assume that a given logical system is consistent. Unfortunately, the latter is difficult to verify and very often is not true. It may happen that addition of a data to a large system makes it inconsistent, and hence destroys the vast amount of meaningful information. We present a logic, called RI (Reasoning with Inconsistency), that treats any set of clauses, either consistent or not, in a uniform way. In this logic, consequences of a contradiction are not nearly as damaging as in the standard predicate calculus, and meaningful information can still be extracted from an inconsistent set of formulae. RI has a resolution-based sound and complete proof procedure. It is a much richer logic than the predicate calculus, and the latter can be imitated within RI in several different ways (depending on the intended meaning of the predicate calculus formulae). We also introduce a novel notion of “epistemic entailment” and show its importance for investigating inconsistency in the predicate calculus.

1. Preface

Most existing logical systems provide no means for reasoning in the presence of inconsistency. For instance, consider the following set of facts: $S = \{\text{flies(tweety)}, \neg\text{flies(tweety)}, \text{grade(john,A)}\}$. Although there is certain amount of inconsistency in $S$, only the knowledge regarding tweety being able to fly is affected. Intuitively, this inconsistency should have no implication regarding John’s grade. However, since $S$ has no model, the standard predicate calculus would warrant any conclusion, including $\text{grade(john,F)}$. On the other hand, checking or guaranteeing consistency in large knowledge-based systems is very expensive, if at all possible. Therefore semantics of such systems cannot be based on the standard logic. Lacking an alternative, many system designers either assume that their systems are always consistent - hardly realistic an assumption, or opt for syntactic approaches to inference, disregarding semantic issues. In the latter case, the user may never be sure regarding the meaning of his program.

Thus, it is desirable to devise such a logic, which we call RI (Reasoning with Inconsistency), in which inconsistency would be not as destructive as in the predicate calculus. The goal of the present paper is to propose just such a logic.

* Work of M. Kifer was supported in part by the NSF grant DCR-8603676, it was partially performed while visiting the Hebrew University of Jerusalem, Israel.
** Work of E.L. Lozinskii was supported in part by Israel National Council for Research and Development under the grants 2454-3-87, 2545-2-87.

2. Introduction

Several researchers have introduced and demonstrated the usefulness of lattice-valued logics. Due to these approaches, each closed formula may be assigned a truth value drawn from a lattice (and sometimes from a bi-lattice) instead of the traditional true /false. Two examples of very popular lattices are given in Figure 1.

Belnap [Bel75, Bel76] has been advocating the use of such logics as a general means for handling inconsistency, while Ginsberg [Gin87] and Sandewal [San85] have shown their utility in explaining the Reiter's results on Default Logic [Rei80], and quite recently Fitting [Fit88a] has developed a fixpoint semantics for bilattice-valued logic programs. With all their elegance, these approaches suffer from a number of drawbacks: The absence of the notion of tautology leads to difficulties in defining refutational proof procedure and to the need in additional complex truth-related notions such as “formula closure”, etc. The notion of logical entailment also differs from a standard one and there is an unpleasant asymmetry in the semantics of implication: normally logic formulas may assume any truth value from the lattice, except for implications, which are strictly 2-valued.

Underlying RI is a lattice of “degrees of belief”, a belief lattice $BL$, with the following properties.

(i) $BL$ contains at least the following four distinguished elements: $t$ (true), $f$ (false), $\top$ (contradiction), and $\bot$ (unknown);

(ii) For every $s \in BL$, $\bot \leq s \leq t$ (is the lattice ordering);
(iii) $\text{lub}(t,f) = \top$; as usual, lub (resp. glb) denotes the least upper (resp. greatest lower) bound.

However, the role of BL in RI is different from that in the lattice-valued logics. In RI the belief lattice is part of the object language rather than the range of truth valuations. Unlike [Bel75, Bel76, Fit88a, Gin87, San85], the RI-interpretations remain 2-valued, the notion of tautology is preserved, and there is a sound and complete resolution-based proof procedure.

Our approach is motivated by a view that the real world is inherently consistent, while inconsistency of its logical description occurs only in the mind of the beholder. So, the real world can tolerate inconsistency of the reasoner’s beliefs, since the latter may not be grounded in the reality. In other words, we distinguish between what the reasoner believes in (at the epistemic level), and what is actually true or false in the real world (at the ontological level). As will be shown in Section 5, RI can faithfully imitate the standard predicate calculus at its ontological level, thus being intolerant to inconsistency, or it can interpret the calculus epistemically being able to tolerate inconsistency in full (or, it can mix the two interpretations). Going back to the comparison with the lattice-valued logics [Bel75, Bel76, Fit88a, Gin87, San85], we note that the latter are completely epistemic. Therefore, they cannot express ontological inconsistency, and, as a result, are unable to fully imitate predicate calculus (see Section 3).

Although RI allows us to reason about consistency of one’s beliefs, it is worth noting that the logic itself is first-order. Some of the ideas about treatment of inconsistency used in RI appeared in [BIS87, BIS89, KIL88]. However, these works are not as general as the present paper, since they are restricted to a subset of logic, and as explained later, consider only a monotonic kind of negation which we call “epistemic”. In the present work, we allow both epistemic and ontological negation, and describe a sound and complete resolution-based theorem-proving procedure for the general logic. We also investigate the relationship of inconsistency in predicate calculus to epistemic and ontological inconsistency in RI.

Elements of our approach can be traced back even further to [BIS87, Sub87, van86]. However, [van86] did not handle inconsistency and negation at all, while [BIS87, Sub87] consider only a subset of logic. There is also an abundance of literature (e.g. [da74, ReB80]) describing other approaches to inconsistency. We are not discussing them here since they employ quite different techniques from those used in this paper.

RI is close in spirit to the logics derived from Post and semi-Post algebras [Eps73, RaE87, Rin88].
However, these logics do not deal with inconsistency and their negation is intuitionistic. In contrast, RI has two negations: onlogic and epistemic, both of which are classical (although it is possible to include an “epistemic intuitionistic negation”, if desired). Furthermore, the Post algebra based logic is less expressive than RI and can be imitated by RI, while the logic of [RaES7] is too expressive to have a resolution-based theorem-proving procedure.

2.1. Syntax of RI

The language of RI is similar to that of predicate calculus. It contains predicates, function symbols, constants, variables, quantifiers, logical connectives, etc. The only syntactical difference is that a literal in our logic is constructed from a literal of predicate calculus by appending to it a suffix which is an element of BL. This suffix represents the truth value of the associated statement according to the reasoner’s belief. Thus, \( p(X) : s \), \( q : t \), and \( \neg r(X) : s \), where \( s, t \in BL \), are literals of RI. Similarly, \( (Y) \) \( r(Y) : t \), \( (X,Y) : s \), and \( (X) : s \) are formulas of RI.

2.2. Semantics of RI

As in the predicate calculus, given an RI language, \( L \), an interpretation, \( I \), is a tuple \( \langle D, I, \Phi \rangle \). Here \( D \) is the domain of \( I \); \( I \) maps any \( k \)-ary function symbol of \( L \) into a mapping \( D^k \rightarrow D \); \( \Phi \) maps any \( m \)-ary predicate symbol into a boolean function \( D^m \rightarrow BL \), \( \rightarrow \{ true, false \} \) such that for every \( p(a) \) there is \( r \in BL \) (depending on \( p \) and \( a \in D^m \) ) s.t. \( I(p(a); s) = true \) if and only if \( s \leq r \) (we are slightly abusing the notation by writing \( I(p(a); s) \) instead of \( I(p(a); s) \) ). These restrictions reflect the view that if the reasoner believes in \( p(a) \) to the degree of \( r \) then he also believes in that to any smaller degree, and also that there is some “maximal” degree of belief in every statement.

A valuation \( v \), assigns values from \( D \) to variables. This mapping can be extended to terms as usual in the first-order logic by combining \( v \) and \( I \): \( v(f(..., s))) = I(f(..., v(s))) \). So, for a term \( t \), \( v(t) \in D \).

For an atomic formula, \( p(t_1, \ldots, t_k) : s \), we write \( I \models p(t_1, \ldots, t_k) : s \) iff \( I(p(v(t_1)), \ldots, v(t_k); s) = true \). Similarly, \( I \models \phi \lor \psi \) iff \( I \models \phi \) or \( I \models \psi \); \( I \models \phi \land \psi \) iff \( I \models \phi \) and \( I \models \psi \); \( I \models \neg \psi \) iff not \( I \models \psi \); \( I \models (\forall X) \psi \) iff \( I \models \psi \) for every \( u \) that may differ from \( v \) only in its \( X \)-value;

We say that \( \phi \) is satisfied by \( I \) iff for every valuation \( v, I \models \phi \). In this case we write \( I \models \phi \).

Following the standard definitions, we say that an interpretation, \( I \), is a model of a set of formulas \( S \) iff every formula \( \phi \) in \( S \) is satisfied in \( I \). A set of formulas \( S \) logically entails a formula \( \phi \), denoted \( S \models \phi \), iff every model of \( S \) is also a model of \( \phi \).

3. Handling Inconsistency in RI

3.1. Epistemic vs. Ontological Inconsistency

Example 1. Consider a set of ground formulas \( S = \{ p : t, q : t, r : t \} \). If one tried to express the same information in the predicate calculus (PC), the most likely result would be \( S^I = \{ p, q, r \} \).

In PC, \( S^I \) is inconsistent, so the standard predicate calculus warrants every conclusion, particularly both \( r \) and \( \neg r \). Intuitively, we may notice the difference in the contribution to the inconsistency made by \( r \) and the first three formulas of \( S^I \). Indeed, the latter are inconsistent regardless of the literal \( r \), that has nothing to do with them.

It is our desire to turn this intuition into a result of a logically correct reasoning. As mentioned earlier, the inconsistency encountered in \( S^I \) accounts for conflicting beliefs of a reasoner, and reflects his contradicting intentions or inadequate information about the real world. Separating the reality from one’s beliefs allows us to tolerate this kind of inconsistency. For instance, this can be done by stating that in \( S^I \), \( p \) and \( q \) are “inconsistent” while \( r \) is just “true”.

To analyze the notion of inconsistency in RI, let us consider the set \( S \) of Example 1, and choose a lattice, say, the one depicted in Figure 1(a). By the definitions in the previous section, \( S \) has several models some of which are listed below:

- \( m_1 \), in which \( p : t, q : t, r : t \) are true;
- \( m_2 \), where \( p : t, q : t, r : t \) are true;
- \( m_3 \), where \( p : t, q : t, r : t \) are true;
- \( m_4 \), where \( p : t, q : t, r : t \) are true.

Actually, \( p : t \) means that the reasoner holds an inconsistent belief regarding \( p \), so here we are dealing with epistemic inconsistency, or e-inconsistency for short. On the other hand, in the real world \( p : t \) and \( \neg p : t \) cannot coexist. Therefore, a set of formulas containing both \( p : t \) and \( \neg p : t \) is ontologically inconsistent (abbr. o-inconsistent). As in PC, o-inconsistent sets of formulas have no model in RI.
An examination of RI models suggests several useful notions. First, we observe that models may differ in the amount of inconsistent beliefs (i.e., e-inconsistency) they contain. For instance, model 2 and 4 contain less e-inconsistency than model 1, and the latter is "more consistent" than model 3. In addition, we may say that model 2 and 4 are "minimal" among the listed models, in the sense that they contain the smallest (w.r.t. the lattice ordering) beliefs about \( p, q, r \). On the other hand, model 2 and 4 have similar amounts of inconsistency.

Formally, we say that an interpretation \( I_1 \) is more (or at least) e-consistent than \( I_2 \) (denoted \( I_2 \sqsubseteq I_1 \)) if for every atom \( p(t_1, \ldots, t_k) \), whenever \( I_1 \models p(t_1, \ldots, t_k) \), it is also the case that \( I_2 \models p(t_1, \ldots, t_k) \) if \( I_1 \) is e-consistent in a class of interpretations, if no interpretation in this class is strictly more e-consistent than \( I_1 \).

Similarly, we say that \( J_1 \) is smaller than \( J_2 \) (denoted \( J_1 \sqsubseteq J_2 \)) if for every atom \( q(t_1, \ldots, t_k) \), whenever \( J_1 \models q(t_1, \ldots, t_k) \), then also \( J_2 \models q(t_1, \ldots, t_k) \) for some \( s \leq r \) in the lattice ordering. An interpretation \( J_1 \) is minimal in a given class of interpretations if this class contains no interpretation strictly smaller than \( J_1 \).

A minimal model may not be most e-consistent, and vice versa, which is illustrated by the following example.

**Example 2.** Consider \( S = \{ p : t, p : f \lor q : t \} \), and suppose \( p \) stands for "it rains" and \( q \) for "take umbrella". \( S \) has two minimal models: \( m_1 \) in which \( p : t \) and \( q : t \) are believed, and \( m_2 \) where \( p : t \) and \( q : f \) are true. Here, \( m_1 \) is strictly more e-consistent than \( m_2 \), and also \( m_1 \) is more appealing to our intuition (i.e., "take umbrella when it rains"). The other model, \( m_2 \), corresponds to the following way of reasoning: "although I was told that it rains and nobody claimed otherwise, I am still sceptic, and draw no conclusion".

Sometimes, certain lattice elements other than \( 1 \) may also be viewed as a kind of inconsistency. For instance, in Figure 1(b), \( a \) (resp. \( d \) ) stands for "concluded true (resp. false) by default". Then the intended meaning of \( d \) would be an "inconsistent default conclusion". Ginsberg [Gin87] suggests that \( d \) should be distinguished from a stronger inconsistency represented by \( 1 \). In this situation, it may be desirable to minimize not just the amount of \( q \) literals, but also other kinds of epistemic inconsistencies (such as \( d \) ).

Formally, let \( \Lambda \subseteq BL \). We write \( I_2 \sqsubseteq I_1 \) (\( I_1 \) is more e-consistent w.r.t. the inconsistencies in \( \Lambda \)) if whenever \( I_1 \models p : \lambda \) for some \( \lambda \in \Lambda \), there is \( \mu \in \Lambda \) s.t. \( \lambda \leq \mu \) and \( I_2 \models p : \mu \). We will not deal with this more general order \( \leq_k \) in this paper.

### 3.2. Ontological vs. Epistemic Negation

The negation, "\( \neg \)" introduced in Section 2 is of an ontological nature, since \( \neg p : \alpha \) is interpreted as the opposite to believing in \( p \) (to the degree of \( \alpha \)). However, at the epistemic level, it is also possible to talk about epistemic negation, denoted \( \sim \), where \( \sim p : t = p : f \) and \( \sim p : f = p : t \).

Formally, epistemic negation, \( \sim \), is a lattice isomorphism \( BL \rightarrow BL \) s.t.

- it is symmetric, i.e., \( \sim \sim \) is an identity mapping, and
- \( \sim t = f \), \( \sim f = t \), \( \sim 1 = 1 \), and \( \sim 1 = 1 \) (the last two equalities actually follow from the monotonicity of lattice homomorphisms).

It is extended to formulae as follows:

(i) \( \sim p : \alpha = p : \sim \alpha \)
(ii) \( \sim \sim \phi = \sim \phi \)
(iii) \( \sim (\phi \lor \psi) \equiv \sim \phi \land \sim \psi \)
(iv) \( \sim (\phi \land \psi) \equiv \sim \phi \lor \sim \psi \)
(v) \( \sim (\forall \chi) \phi \equiv (\exists \chi) \sim \phi \)

In PC, the usual implication \( \phi \rightarrow \psi \) is defined as \( \phi \lor \sim \psi \). In RI we have a choice between the ontological implication: \( \phi \rightarrow \psi \equiv \sim \phi \lor \psi \), and the epistemic implication: \( \phi \rightarrow \psi \equiv \sim \phi \lor \psi \).

The two kinds of implication differ significantly in their properties, especially in the way they propagate inconsistent beliefs.

**Example 3.** Consider \( S = \{ q : t \rightarrow p : t; q : t \} \). It is easy to see that \( S \models p : t \). Furthermore, even if \( q \) were an inconsistent belief, \( p : t \) would still follow: \( \{ q : t \rightarrow p : t; q : t \} \models p : t \). Thus, ontological implication allows to draw conclusions from inconsistent beliefs. This corresponds to the following way of reasoning: \( q \) is an inconsistent belief, but in the real world either \( q \) or \( \neg q \) is true. So, to ensure that \( q : t \rightarrow p : t \) is true in every case, we must assume \( p : t \).

For comparison, let us replace \( \rightarrow \) by \( \sim \) in the above example: \( T := \{ q : t \sim p : t; q : t \} \). Surprisingly, it is no longer true that \( T \models p : t \). Indeed, \( q : t \sim p : t \) is equivalent to \( q : t \lor p : t \), and there is a model \( I \) in which \( I \models q : t \), but only \( I \models p : t \) (particularly, \( I \models p : t \)).

We thus see that the epistemic implication is "overly cautious". Not only does it refuse to draw conclusions from inconsistent beliefs, but also it...
guards from the possibility that one of the premises (e.g., \( q \) in Example 3) may somehow turn out to be inconsistent after all. This excessive cautiousness can be countered with a kind of the closed world assumption applied to e-inconsistency, which suggests preferring models with the least amount of e-inconsistency.

In other words, in dealing with the epistemic implication we propose to narrow the attention to the set of most e-consistent models. Logical entailment restricted to the most e-consistent models will be called epistemic entailment, denoted \( \models _{ep} \).

It is easy to see that in Example 3, \( T \models _{ep} p : t \). While countering the overly cautious behaviour of \( \models \), epistemic entailment still does not permit drawing conclusions from inconsistent premises, i.e., if \( T' = \{ q : t \rightarrow p : t; q : t \} \), then \( T' \not\models _{ep} p : t \), since the model, \( I \), of \( T' \) in which \( I \models q : t \), only \( I \models p : t \), is most e-consistent and falsifies \( p : t \).

RI with its two kinds of negation and implication provides a rich framework for modeling many different situations requiring handling of inconsistency. For instance, Belnap [Be175, Be176] takes a stand that inconsistent beliefs should not be used in the inference and defines implication accordingly. As we have shown, our epistemic implication corroborates that view. On the other hand, Touretski, Horty, and Tho-

mason [THT87] argued that in some situations inconsistency may be propagated through implications, which corresponds to the ontological implication in RI.

The distinction between the ontological and epistemic versions of negation and implication sheds additional light on the logics used in [BIS87, BIS89, Kil88, Sub87, van86] from which RI was gradually evolving. It turns out that those logics use the ontological implication, but the negation is epistemic ([van86] does not consider negation). They also use special kinds of lattices and [BIS87, Sub87, van86] are subsumed both by [Kil88] and RI. On the other hand, [Kil88] cannot be directly compared to RI, since the former uses quantified variables and evaluable monotonic functions over BL.

4. Refutational Proof Procedures

An appealing feature of RI is that the standard techniques, such as skolemization, Herbrand's theorem, refutational proof procedure, etc. are still applicable. We assume that the reader is familiar with those standard notions of first order logic, and present their counterparts in RI.

Skolemization procedure in RI is identical to that of PC, and we obtain the following result whose proof is identical to the proof of Skolem theorem in predicate calculus.

**Theorem 1** [cf. Skolem Theorem] Let \( S \) be a set of formulas, and \( \phi \) - a formula. Let \( S \cup \{ \neg \phi \} \) o-inconsistent and \( \phi \) respectively. Then \( S \cup \{ \neg \phi \} \) is o-inconsistent iff \( S \cup \{ \neg \phi \} \) is.

Given a language, \( L \), the Herbrand universe in RI is the same as that in PC. Similarly, the Herbrand base is the set of all ground atoms of \( L \) (but recall the suffixes). A Herbrand interpretation, \( I \), is a subset of the Herbrand base s.t. for every ground literal \( p \) (without the suffix) there is \( R \in BL \) (depending only on \( p \)) s.t. \( p : s \in I \) if and only if \( s \leq r \).

In general, Herbrand's theorem does not hold for RI as follows from the following example.

**Example 4:** Consider a belief lattice obtained from the 4-valued lattice (Figure 1a) by filling in the edges of the diamond with a continuum of values. For definiteness, assume that the continuum corresponding to the edge \( < t_1, t_2 > \) is linearly ordered between \( t_1 \) and \( t_2 \) and has the form \( \{ t_r | r \in [0,1] \} \), where \( t_0 = 0 \) and \( t_1 = 1 \). Let \( S \) consist of \( \neg p : t_1 \) and the set \( \{ p : t_r | 0 < r < 1 \} \).

Clearly \( S \) is unsatisfiable, for in every model of \( S \) \( p : t_1 \equiv \neg p : t_1 \) must be true. However, every finite subset of \( S \) is satisfiable, since \( t_1 \) is not a limit of any finite subset of \( \{ t_r | 0 < r < 1 \} \).

Nevertheless, Herbrand's theorem holds in several important special cases which include logic programs and the situations when BL is finite.

**Theorem 2** (cf. Herbrand's theorem) Consider a set of possibly nonground clauses, \( S \), which involve only a finite number of different belief suffixes (this condition is always satisfied when BL is finite or when its structure is finite, e.g., a logic program). Then \( S \) is o-inconsistent iff so is some finite subset of its ground instances.

The notion of substitution in RI is the same as in PC. There is a slight difference in unification. A pair of atoms, \( p (t_1, \ldots, t_k) : s \) and \( p (t_1', \ldots, t_k') : r \), is unifiable iff there exists a unifier of \( p (t_1, \ldots, t_k) \) and \( p (t_1', \ldots, t_k') \).

Binary resolution is defined as follows. Let \( p (t_1) : s \lor \phi \) and \( \neg p (t_1') : r \lor \psi \) be a pair of clauses not sharing common variables such that \( \theta = \text{ugu}(p (t_1), p (t_1')) \) and \( s \geq r \) (note the asymmetry!). Then their resolvent is \( (\phi \lor \psi) \).

Next, consider a clause \( p (t_1) : s_1 \lor \ldots \lor p (t_k) : s_k \lor \phi \). Then \( p (t_1) : \theta \geq \text{glb}(s_1 \lor \phi) \lor \theta \lor \phi \), where \( \theta = \text{mgl}(p (t_1), \ldots, p (t_k)) \) is a factor of the clause. Similarly, a factor of \( \neg p (t_1) : s_1 \lor \ldots \lor \neg p (t_k) : s_k \lor \phi \) is \( \neg p (t_1) : \theta \land \text{lub}(s_1 \lor \phi) \), where \( \theta \) as before.
Notice that unlike PC, there is an asymmetry in the definition of resolution, since the positive literal to be resolved upon has not to be believed in weaker than the negative one. More important, unlike PC, resolution and factorization alone are not complete as derivation rules. For instance, these rules do not suffice to derive an empty clause from $\neg p \lor \neg q \lor m$, although this set is unsatisfiable. Indeed, to be a model of $S$, $I$ has to contain $p : f$ and $p : t$. By the definition of Herbrand models given above, $I$ has to contain $p : \text{true}$ and $m : \text{true}$. But then $I$ falsifies $\neg p : i$. Furthermore, $p : i$ for any atom $p$ is a tautology in RI, yet it is not derivable by the above rules. To complete the picture we introduce two more rules, called reduction and elimination.

Given a pair of clauses $p(i) : s \lor \phi$ and $p(i') : t \lor \psi$ such that $\theta = \text{mgu}(p(i), p(i'))$, their reduction is the clause $p(t) : \text{lub}(s\lor t) \lor \phi \lor \psi$. Coming back to the example in the previous paragraph, we can reduce $p : f$ and $p : t$, obtaining $p : i$. The latter literal can then be resolved with $\neg p : i$, yielding the empty clause. The elimination rule being applied to a clause $C$ simply removes all literals in $C$ of the form $\neg p(i) : i$. The resulting clause is called an eliminant of $C$.

**Lemma 1.** Resolution, factorization, elimination, and reduction are sound derivation rules. [ ]

Given a set of clauses $S$, its refutation is a derivation from $S$ of an empty clause using resolution, factorization, elimination, and reduction.

**Theorem 3.** Refutation is a sound and complete procedure for testing o-inconsistency of sets of clauses involving only a finite number of different belief constants. That is, if $S$ is such a set then it is o-inconsistent if there is a refutation of $S$. [ ]

**Example 5.** Consider the following set of clauses:

- (i) $p : i \lor q : i$
- (ii) $p : t \lor \neg q : t$
- (iii) $\neg p : i$
- (iv) $p : f$

The following refutation illustrates the basic concepts.

- (v) $p : i \lor p : t$ (resolvent of (i) and (ii));
- (vi) $p : t$ (factor of (v));
- (vii) $p : i$ (reduction of (vi) and (iv));
- (viii) empty clause (resolvent of (vii) and (iii)).

**5. Relationship to Predicate Calculus**

As mentioned earlier, the purpose of RI is to cope with inconsistency. For consistent systems it should be expected to yield essentially the same consequences as the standard predicate calculus. However, since RI is a richer logic than PC, the latter can be interpreted in RI in several different ways. In this section we present two different natural embeddings of PC into RI and discuss their properties.

The first, epistemic embedding, $\mathbb{E}_{epi}$, views formulas of PC as beliefs and interprets negation, $\neg p$, in a rather restricted sense: as a belief in the falsehood of $p$. Formally, $\mathbb{E}_{epi}(p(...)) = p(...): t$ and $\mathbb{E}_{epi}(\neg p(...)) = p(...): f$ ($\equiv \neg p(...): t$). The second, ontologic embedding, $\mathbb{E}_{ont}$, interprets negation $\neg p$, in a much stronger sense: as the opposite to believing in $p$. Thus, in contrast to the epistemic embedding $\mathbb{E}_{epi}$, $\mathbb{E}_{ont}$ views $S$ not as a collection of beliefs, but ontologically, as knowledge about what the reasoner believes in or does not. Formally, $\mathbb{E}_{ont}(p(...)) = p(...): t$ and $\mathbb{E}_{ont}(\neg p(...)) = \neg p(...): t$. Each embedding extends to formulas by allowing it to commute with quantifiers, $\forall$, and $\exists$.

Closely related to these embeddings is a mapping, $\Xi$, of PC-interpretations into the set of RI-interpretations. Let $I$ be a Herbrand interpretation of PC. Then $\Xi(I)$ is an interpretation of RI s.t. for each ground atom, $p$, $I \models p$ iff $\Xi(I) \models p : t$ and $I \not\models p$ iff $\Xi(I) \models p : f$.

More generally, let $I = (D, \Gamma, \Phi)$ be a (not necessarily Herbrand) interpretation of PC, where $\Phi$ interprets predicate symbols as relations. Then $\Xi(I) = (D, \Gamma', \Phi')$, where for each predicate symbol, $p$, and $a_1, \ldots, a_n \in D$, $\Phi'(p(a_1, \ldots, a_n) : t) = \text{true}$ iff $\Phi(p(a_1, \ldots, a_n)) = \text{true}$ and $\Phi'(p(a_1, \ldots, a_n) : f) = \text{true}$ iff $\Phi(p(a_1, \ldots, a_n)) = \text{false}$.

As seen from the definition, $\Xi$ yields a special kind of interpretations of RI: those in which each atom is either known to be true or false (but not both). We say that an RI-interpretation, $I = (D, \Gamma, \Phi)$, is a $\Gamma$-interpretation if for each $n$-ary predicate symbol, $p$, and $a_1, \ldots, a_n \in D$, either $\Phi(p(a_1, \ldots, a_n) : t) = \text{true}$ or $\Phi(p(a_1, \ldots, a_n) : f) = \text{true}$, but not both. For Herbrand interpretations it means that for every ground atom $p$, either $p : t \in I$, or $p : f \not\in I$, but not both.

It is easy to verify that for each PC interpretation, $I$, $\Xi(I)$ is a $\Gamma$-interpretation. Furthermore, for a given PC language, $\Lambda$, and a lattice, $BL$, let $\Lambda_{BL}$ denote the corresponding language of RI. Then $\Xi$ is a 1-1 mapping from interpretations of $\Lambda$ onto the set of $\Gamma$-interpretations of $\Lambda_{BL}$. The inverse mapping, $\Upsilon$: $\{ \text{interpretations of } \Lambda_{BL} \} \rightarrow \{ \text{interpretations of } \Lambda \}$, is defined as follows: Let $I^{\Gamma} = (D, \Gamma, \Phi')$ be a $\Gamma$-interpretation of $\Lambda_{BL}$. Then $\Upsilon(I^{\Gamma}) = I$, where $I = (D, \Gamma, \Phi)$ interprets constants and function symbols of $\Lambda$ the same way as $I^{\Gamma}$, and $\Phi(p(a_1, \ldots, a_n)) = \text{true}$ iff $\Phi'(p(a_1, \ldots, a_n) : t) = \text{true}$ and $\Phi(p(a_1, \ldots, a_n)) = \text{false}$ iff $\Phi'(p(a_1, \ldots, a_n) : f)$.
Theorem 4. Let $S$ be a set of formulas in PC. Let $M(S)$ denote the set of models of $S$, $M_{tf}^{in}(S)$ - the set of all tf-models of $\Xi_{out}(S)$, and $M_{tf}^{incl}(S)$ -the same for $\Xi_{incl}(S)$. Then

1. $M_{tf}^{incl}(S) = M_{tf}^{in}(S)$ (hereafter denoted just $M_{tf}(S)$).
2. $\Xi : M(S) \rightarrow M_{tf}(S)$ is a 1-1 mapping onto $M_{tf}(S)$.
3. $S \models \phi$ if $M \models \Xi_{incl}(\phi)$ (resp. $M \models \Xi_{out}(\phi)$) for every $M \in M_{tf}(S)$.

Informally, Theorem 4 says that for consistent sets of formulae, RI-logic does not buy us much new: both embeddings, $\Xi_{incl}$ and $\Xi_{out}$, yield essentially the same results, and the set of PC-models of $S$ can be naturally identified with a representative subclass, $M_{tf}(S)$, of models of $\Xi_{out}(S)$ and $\Xi_{incl}(S)$.

However, when $S$ is inconsistent, then both $M(S)$ and $M_{tf}(S)$ are empty, and this is one of the situations when the expressive power of RI becomes useful.

Theorem 5: A set of PC formulae, $S$, is inconsistent if and only if $\Xi_{out}(S)$ is o-inconsistent. 

According to Theorem 5, $\Xi_{out}$ is a faithful imitation of PC inside RI in the following sense:

Corollary 1: $S \models \phi$ in PC iff $\Xi_{out}(S) \models \Xi_{out}(\phi)$ in RI.

Particularly, $\Xi_{out}$ does not let one reason about inconsistency in PC, which is not surprising, since this mapping regards $S$ as a collection of ontologically correct statements about reasoner’s beliefs (which cannot be inconsistent), rather than as an “internal state” of reasoner’s beliefs (which may be inconsistent).

As we have already mentioned, existing logical systems are too sensitive to contradicting information. It may happen that addition of an elementary data (like a unit clause) to a large system $S$ makes it inconsistent, and hence destroys the meaningful information contained in $S$. With this in mind, one of the goals of this work is showing that RI is just such a logic that can isolate the information which humans would reasonably label as the “cause of inconsistency” from the data which intuitively has nothing to do with the inconsistency, and therefore should preserve its meaning. In the rest of this section we show that the epistemic interpretation of PC within RI, $\Xi_{incl}$, results in the desired isolation of inconsistency.

It is easy to see that, every set, $R$, is mapped by $\Xi_{incl}$ into an o-consistent set $R'$. Thus, $\Xi_{incl}$ permits full tolerance of inconsistency of beliefs. By Theorem 4, if $R$ is inconsistent, then $R'$ has no tf-model. Thus, in this case every model of $R'$ necessarily contains literals with suffix $\vdash$. Therefore, inconsistency in $R$ is reflected by the occurrences of $\vdash$ in the models of $R'$. Moreover, an appearance of a $p \vdash$ in a most o-consistent model of $R'$ indicates that $p$ may be a reason for inconsistency in $R$. We formalize this observation below.

In the ensuing discussion we will restrict ourselves to the case of formulae in skolemized form. We do not have a characterization of the cause of inconsistency in the general case. However, this restriction seems satisfactory for most of the practical purposes, since skolemization is routinely performed in resolution-based inconsistency checking procedures. Once skolemized, we can assume w.l.o.g. that the formulae are in clausal form. Since, by Herbrand’s Theorem, any inconsistent set of clauses has an inconsistent ground (even finite) instance, we can further limit our discussion to the case of ground clauses.

Consider an inconsistent set, $R$, of ground clauses in PC. A ground atom, $p$, is a suspect w.r.t. $R$ (a suspected cause of inconsistency in $R$) if there are consistent subsets $R_{1}$, $R_{2} \subseteq R$ s.t. $R_{1} \models p$ and $R_{2} \models \neg p$. Any set of formulae, $E$, s.t. $R_{1} \cup R_{2} \subseteq E \subseteq R$ (particularly, $R$ itself) is called an indictment against $p$ w.r.t. $R$.

The notion of a “suspect” does not suffice to characterize the cause of inconsistency. To see why, consider the following example. Let $S$ be \{ $p$, $\neg p$, $q$, $p \rightarrow q$ \}. Then $S$ indicted both $p$ and $q$. However, intuitively, $p$ seems more of an inconsistency cause than $q$, since $p$ is suspected regardless of what are the suspicions about $q$. In other words, removing clauses involving $q$ does not make $S$ consistent because $p$ and $\neg p$ can still be derived. The following definitions formalize this intuition.

A set of suspects, $S$, implicates a suspect, $p$, if for every indictment, $E$, against $p$ there is $q \in S$ indicted by $E$. A gang (of culprits of inconsistency in $R$) is a set, $G$, of suspects w.r.t. $R$ s.t.

1. every suspect w.r.t. $R$ is implicated by $G$ and
2. no proper subset of $G$ possesses property (1).

There may exists several gangs w.r.t. $R$. The following example illustrates these notions.

Example 6: Let the set of formulae be \{ $p$, $q$, $r$, $\neg q \rightarrow p$, $\neg r \rightarrow q$, $\neg p \rightarrow r$ \}. It is easy to verify that there are three gangs: \{ $p$, $q$ \}, \{ $q$, $r$ \}, and \{ $r$, $p$ \}. The set \{ $p$, $\neg q \rightarrow p$ \} is an indictment against $q$ and $p$, the set \{ $q$, $\neg r \rightarrow q$ \} is an indictment against $q$ and $r$, etc.
For another example, consider the set of formulae \( R = \{ q, \neg q, r, \neg r, p \rightarrow q, p \rightarrow r \} \). Here the only gang is \( \{ q, r \} \). Although \( p \) is a suspect, it is implicated by the gang, which can be verified, say, by considering all the indictments against \( p \).

For instance, consider the following indictment against \( p : \{ q, r, q \rightarrow p, r \rightarrow \neg p \} \). This is also an indictment against \( q \) and \( r \). To see this in case, say, of \( q \), consider \( R_1 = \{ q \} \) and \( R_2 = \{ r, q \rightarrow p, r \rightarrow \neg p \} \). Clearly, both \( R_1 \) and \( R_2 \) are consistent and \( R_1 \models q \) while \( R_2 \models \neg q \).

It is interesting to note that our treatment of the cause of inconsistency follows the standard path of defining logical entailment. The notion of an "indictment" corresponds to the concept of a model in logical theories. The notion of suspect implication then becomes identical to logical entailment once the term "model" is replaced by the term "indictment". Thus, a "gang" is a minimal set of suspects which "logically entail" in that sense any other suspect. This intuition is strengthened by the following theorems, which say that under \( \Xi \) only the gang members (the gangsters) may be mapped into e-inconsistent literals of RI, and on the other hand, information which does not cause inconsistency preserves its meaning. Particularly, by (1) and (2) of Theorem 6 below, if \( S \) is consistent in PC, then most models of \( \Xi (S) \) are also e-consistent. In other words, \( \Xi \) is a faithful translation of the cause of inconsistency in PC into e-inconsistency in RI.

**Theorem 6:** Let \( S \) be a set of ground clauses. Then the following holds true.

1. If \( G \) is a gang then there is a most e-consistent model \( M \) of \( S \) such that \( G = \{ p \mid M \models p : 1 \} \).
2. Let \( M \) be a most e-consistent model of \( S \). Then the set \( \{ p \mid M \models p : 1 \} \) is a gang.

For instance, in Example 1 we have \( S = \Xi (\{ \neg q \}) \), and \( \{ p \} \) and \( \{ q \} \) are gangs. Thus, every model of \( S \) (cf. \( m_1, ... , m_4 \)) contains either \( p \) or \( q \). In contrast, \( r \) is not a cause of inconsistency of \( S \), and, indeed, \( r : 1 \) does not belong to any most e-consistent model (namely, \( m_2, m_3, m_4 \)) of \( S \).

Our next result shows that \( \Xi \) is a well-defined sense, is able to "salvage" that part of the information which can be viewed as undamaged by the inconsistency. First, we need a number of definitions.

Let \( S \) be, as before, a (possibly infinite) set of ground clauses. Consider a literal, \( l \), of the form \( p \) or \( \neg p \), where \( p \) is a ground atomic formula. We say that \( l \) is spoiled by the inconsistency in \( S \), if \( p \) belongs to every gang; it is supported if there is a consistent subset \( S' \) of \( S \) which entails \( l : S' \models l \).

In this case we say that \( S' \) is an evidence for \( l \).

The literal \( l \) is recoverable if either

(i) it is supported, not spoiled, and for every gang, \( G \), in \( S \), either \( p \in G \) (recall: \( l \in \{ p, \neg p \} \) ), or there is an evidence, \( S' \), for \( l \), s.t. \( S' \) mentions no gangster from \( G \), or

(ii) \( l \) is a negation of a literal, \( l' \), which satisfies (i) (i.e., negation of a recoverable literal is also recoverable).

Finally, a supported literal is damaged (by the inconsistency) if it is neither spoiled nor recoverable.

We will see that the spoiled literals cannot be assigned any meaning other than that of an inconsistent belief. Recoverable literals, as their name suggests, can be viewed as the "robust" part of the information in \( S \); they preserve their meaning even though this meaning is "obscured" by the inconsistency. Intuitively, the robustness of recoverable literals comes from the fact that their supporting evidences are not based on any specific gang, which makes those evidences sufficiently credulous.

On the other hand, a damaged literal should be viewed as a literal which may have had a meaning before \( S \) became inconsistent (since it still has supporting evidences), but that meaning cannot be reliably recovered, because these evidences are obtained from the same gang. For instance, in \( \{ q, \neg q, q \rightarrow p, \neg p, q \rightarrow r, q \rightarrow \neg r \} \), \( q \) is spoiled, \( p \) and \( \neg p \) are recoverable, and \( r \), \( \neg r \) are damaged. The ability to recover the meaning of \( \neg p \) can be intuitively explained by the fact that inconsistency in \( q \) precludes deriving \( p \), which leaves us with \( \neg p \). Informally, \( \neg p \) can be viewed as a true statement "despite the inconsistency". Its negation, \( \neg \neg p \), can then be thought of as a false statement. The next theorem justifies our definitions.

**Theorem 7:** Let \( S \) be as before and \( l \) be a literal of the form \( p \) or \( \neg p \), where \( p \) is a ground atom. Then

1. \( l \) is spoiled iff \( \Xi (S) \not\models p : 1 \).
2. \( l \) is recoverable iff \( \Xi (S) \not\models \neg p : 1 \) and either \( \Xi (S) \models \Xi (l) \) or \( \Xi (S) \models \neg \Xi (l) \).

Finding the recoverable part of the information is at least as hard as the logical entailment. This is because if \( S \) is consistent, recoverability reduces to logical entailment. In general, finding whether a literal is recoverable requires a proof procedure for epistemic entailment. At present, we only have such procedure for a special case when \( S \) is a logic.

---

1. Since \( l \) is not spoiled, there is at least one such evidence.
Corollary 2: A ground literal \( l \) is spoiled in \( S \) if both \( l \in S \) and \( \neg l \in S \).

6. Connection to Nonmonotonic Reasoning and Logic Programming

As is well known, nonmonotonic reasoning (e.g. [McC86, Rei80]) extends significantly the power of PC. It can be incorporated in RI along the same lines, and will be reported in a forthcoming work. However, even without using circumscription or defaults, RI yields more informative results than PC.

Example 7. Consider the following system:

\[
\begin{align*}
\text{flies}(X) & \leftarrow \text{bird}(X) \\
\neg \text{flies}(X) & \leftarrow \text{penguin}(X) \\
\text{bird}(\text{tweety}) & \\
\text{bird}(\text{fred})
\end{align*}
\]

Both PC and RI (under \( \Xi_{epi} \) embedding) conclude \( \text{flies}(\text{tweety}) \) and \( \text{flies}(\text{fred}) \). However, adding \( \text{penguin}(\text{tweety}) \) makes the system PC-inconsistent precluding any informative conclusion. In contrast, under the embedding \( \Xi_{epi} \), RI discovers the inconsistency regarding \( \text{flies}(\text{tweety}) \) (\( \text{flies}(\text{tweety}) : 1 \)), but still infers \( \text{flies}(\text{fred}) : t \).

Although recent advances in Logic Programming [Min88] provided an adequate semantics for negation in rule bodies, still negation is disallowed in rule heads. Particularly, rather natural situations such as that of Example 7 do not have a straightforward representation. In the framework of RI such programs can be given a natural and computationally realizable semantics. Additionally, it now becomes possible to exploit the two kinds of implications found in RI.

In Logic Programming, negation in rule bodies is normally treated by a form of closed world assumption. An appropriate semantics for this treatment was recently proposed in [ABW88, Li88, Pr88]. Inferences made according to such semantics may not always be logically valid, but are rather derived by default. Several researchers argued that preserving the distinction between true and default facts may be useful [Gin87, Per87]. It can be shown that in RI, keeping track of the defaults is easy, if one uses a lattice, BL, such as the one in Figure 1(b).

A number of recent proposals allow negated literals in rule heads [BIS87, BIS89, Fit88b, Kil88]. However, all these works consider only epistemic negation. In addition, the treatment of negation in [BIS87, BIS89, Fit88b] is monotonic, which precludes default reasoning. Recently Kifer and Li [Kil88] have shown that for a special type of lattices, epistemic negation can be also treated non-monotonically. This result was later extended to the case when BL is a bilattice with conflation [Kil89]. We believe that allowing both types of negation and non-monotonicity, epistemic and ontological, results in a very rich logic programming language, which will be discussed in a separate work.

7. Conclusion

We presented a logic capable of handling inconsistent beliefs. Several extensions to RI are possible. First, without any difficulty one can associate different lattices to different predicates. This may be useful when granularity of degrees of beliefs differs from predicate to predicate.

Second, it is possible to relax the restriction that in a literal, \( p : s \), the suffix \( "s" \) is a lattice constant. Instead, a number of monotonic lattice functions and lattice variables could be allowed, which brings in the terms of the form \( p : f(X,Y,a) \), etc. Lattice variables can be quantified, which adds more power to the language, for instance, allowing rules like

\[
\forall X \forall Y (\text{flies}(X) : f(Y) \leftarrow \text{bird}(X) : Y).
\]

Unfortunately, we are not aware of any complete refutation procedure for such an extended logic. The difficulty here is that it would be unreasonable to assume that lattice functions are uninterpreted (as in the theory of Logic Programming), since for the most part lattices are finite. In [Kil88, Kil89] a complete proof procedure was developed for a restricted logic in which only Horn-like rules are permitted, and body literals may have only lattice variables and constants as suffixes. The same ideas can be adapted for a subset of RI.

In a broader perspective, RI can be regarded as a technique for extending the scope of many "conventional" logics (including modal logics) to dealing with inconsistency. In this sense, the present work is an example of applying that technique to the standard predicate calculus.

Acknowledgements: Joseph Tashjian has found several inaccuracies in the earlier draft of this paper, and Anita Wasilewska pointed out the connection to Post algebras. We also thank V.S. Subrahmanian for the pointers to several papers on paraconsistent logics.

\[2\] The actual lattice structure appropriate for tracking down default inferences is problem-dependent and may be quite different from that of Figure 1(b).
8. References


