Approximate Reasoning and Paraconsistency - Preliminary Report

Marcelo Finger, *Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Brazil, mfinger@ime.usp.br*

Renata Wassermann, *Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Brazil, renata@ime.usp.br*

Abstract

This paper relates two apparently distinct topics in non-classical logics. One of the main criticisms of classical logic is that theories trivialize in the presence of inconsistency. Paraconsistent logics were introduced in the fifties by da Costa to avoid such problem. Da Costa's system C_1 avoids trivialization by changing the behavior of negation.

Cadoli and Schaerf have proposed the Calculus S_3, where approximate entailment is used as a way of reaching at least partial results when solving a problem completely would be too expensive. However, the language they use is restricted to that of clauses, i.e., negation appears only in the scope of atoms and there is no implication.

In this paper, we will relate a formalism developed by computer scientists aiming at computational efficiency (Cadoli and Schaerf's system S_3) and another developed by logicians aiming to deal with inconsistency (da Costa's C_1). We propose a proof method for S_3 to use with C_1. We also give a sound and complete axiomatization for S_3, based on the axiomatization of the calculus C_1, thus sowing how similar they are, and where they differ.

1 Introduction

Paraconsistent logics were introduced by da Costa in the end of the fifties to avoid a well known problem of classical logic, that of trivialization in the presence of inconsistency. If we happen to have both a formula and its negation in a set, classical logic can infer from this set just any formula of the language. This is not very reasonable if we think of realistic systems storing information. Suppose we update, by mistake, a database with inconsistent information about a product in a shop. It does not seem reasonable to say that now all information about other products is "contaminated" by the inconsistency. The inconsistency damages the database only locally, i.e., only for reasoning about that particular product.

Da Costa's system C_1 avoids trivialization by changing the behavior of negation. Some formulas are said to be *well behaved* and for them, negation behaves classically. For formulas which are not well behaved, it may happen that both the formula and its negation are assigned the value true.

Logic has been used in several areas of Artificial Intelligence as a tool for representing knowledge as well as a tool for problem solving. One of the main criticism to the use of logic as a tool for automatic problem solving refers to the computational complexity of logical problems. Even if we restrict ourselves to classical propositional logic, deciding whether a set of formulas logically implies a certain formula is an NP-complete problem [GJ79].

Cadoli and Schaerf have proposed the use of approximate entailment as a way of reaching at least partial results when solving a problem completely would be too expensive [SC95]. Their method consists in defining different logics for which satisfiability is easier to compute than classical logic and treat these logics as upper and lower bounds for the classical problem. In [SC95], these approximate logics are defined by means of valuation semantics and algorithms for testing satisfiability. The language they use is restricted to that of clauses, i.e., negation appears only in the scope of atoms and there is no implication.

The approximations are based on the idea of a context set S of atoms. The atoms in S are the only ones whose consistency is taken into account in the process of verifying whether a given formula is entailed by a set of formulas. As we increase the size of the context set S, we get closer

to classical entailment, but the computational complexity also increases.

In this paper, we will relate a formalism developed by computer scientists aiming at computational efficiency (Cadoli and Schaerf's system) and another developed by logicians aiming to deal with inconsistency (da Costa's C_1).

As a result, we can adapt the proof method that we have developed for S_3 to use with C_1. We also give a sound and complete axiomatization for S_3, based on the axiomatization of the calculus C_1. In Cadoli and Schaerf's work, the treatment of S_3 is purely semantical. An axiomatization was still missing.

The paper proceeds as follows: we briefly present Cadoli and Schaerf's S_3 system for approximate entailment. Then we present an extended version of S_3 which we call *Fine*S_3. We then present da Costa's C_1 and show some results about the relationship between *Fine*S_3 and C_1.

1.1 Preliminaries

Let $\mathcal{P}$ be a countable set of propositional letters. We concentrate on the classical propositional language $\mathcal{L}_C$ formed by the usual boolean connectives $\rightarrow$ (implication), $\wedge$ (conjunction), $\vee$ (disjunction) and $\neg$ (negation).

Throughout the paper, we use lowercase Latin letters to denote propositional letters, lowercase Greek letters to denote formulas, and uppercase letters (Greek or Latin) to denote sets of formulas.

A *propositional valuation* v_p is a function $v_p : \mathcal{P} \rightarrow \{0, 1\}$.

2 Approximate Entailment

We briefly present here the notion of *approximate entailment* and summarize the main results obtained in [SC95].

Schaerf and Cadoli define two approximations of classical entailment: $\models_S^1$ which is complete but not sound, and $\models_S^3$ which is sound and incomplete. These approximations are carried out over a set of atoms $S \subseteq \mathcal{P}$ which determines their closeness to classical entailment. In the trivial extreme of approximate entailment, i.e., when $S = \mathcal{P}$, classical entailment is obtained. At the other extreme, when $S = \emptyset$, $\models_S^1$ holds for any two formulas (i.e., for all α,β, we have $\alpha \models_S^1 \beta$) and $\models_S^3$ corresponds to Levesque's logic for explicit beliefs [Lev84], which bears a connection to relevance logics such as those of Anderson and Belnap [AB75].

In an S_1 assignment, if $p \in S$, then p and $\neg p$ are given opposite truth values; if $p \notin S$, then p and $\neg p$ both get the value 0. In an S_3 assignment, if $p \in S$, then p and $\neg p$ get opposite truth values, while if $p \notin S$, p and $\neg p$ do not both get 0, but may both get 1. The set of formulas for which we are testing entailments is assumed to be in clausal form. Satisfiability, entailment, and validity are defined in the usual way.

The following examples illustrate the use of approximate entailment. Since $\models_S^3$ is sound but incomplete, it can be used to approximate $\models$, i.e., if for some S we have that $B \models_S^3 \alpha$, then $B \models \alpha$. On the other hand, $\models_S^1$ is unsound but complete, and can be used for approximating $\not\models$, i.e., if for some S we have that $B \not\models_S^1 \alpha$, then $B \not\models \alpha$.

EXAMPLE 2.1 ([SC95])
We want to check whether $B \models \alpha$, where $\alpha = \neg$`cow` $\vee$ `molar-teeth` and

$B =$ {$\neg$`cow` $\vee$ `grass-eater`,
`¬dog`$\vee$ `carnivore`,
$\neg$`grass-eater` $\vee$ $\neg$`canine-teeth`,
$\neg$`carnivore` $\vee$ `mammal`,
$\neg$`mammal` $\vee$ `canine-teeth` $\vee$ `molar-teeth`,
$\neg$`grass-eater` $\vee$ `mammal`,
$\neg$`mammal` $\vee$ `vertebrate`,
$\neg$`vertebrate` $\vee$ `animal`}.

For $S = \{\texttt{grass-eater}, \texttt{mammal}, \texttt{canine-teeth}\}$, we have that $B \models^3_S \alpha$, hence $B \models \alpha$.

EXAMPLE 2.2 ([SC95])
We want to check whether $B \not\models \beta$, where $\beta = \neg\texttt{child} \vee \texttt{pensioner}$ and

$$\begin{aligned} B = \ & \{\neg\texttt{person} \vee \texttt{child} \vee \texttt{youngster} \vee \\ & \qquad \vee \texttt{adult} \vee \texttt{senior}, \\ & \neg\texttt{pensioner} \vee \texttt{senior}, \\ & \neg\texttt{youngster} \vee \texttt{student} \vee \texttt{worker}, \\ & \neg\texttt{senior} \vee \texttt{pensioner} \vee \texttt{worker}, \\ & \neg\texttt{adult} \vee \texttt{student} \vee \texttt{worker} \vee \\ & \qquad \vee \texttt{unemployed}, \\ & \neg\texttt{pensioner} \vee \neg\texttt{student}, \\ & \neg\texttt{student} \vee \texttt{child} \vee \texttt{youngster} \vee \texttt{adult}, \\ & \neg\texttt{pensioner} \vee \neg\texttt{worker}\}. \end{aligned}$$

For $S = \{\texttt{child}, \texttt{worker}, \texttt{pensioner}\}$, we have that $B \not\models^1_S \beta$, and hence $B \not\models \beta$.

Note that in both examples above, S is a small part of the language. The approximation of classical inference is made via a simplification of the belief base B as follows (for a given conclusion α and context set S):

LEMMA 2.3 ([SC95])
Let *simplify-1*(B, S) be the result of deleting all literals of B which mention atoms outside S. B is S_1-satisfiable if and only if *simplify-1*(B, S) is classically satisfiable.

LEMMA 2.4 ([SC95])
Let *simplify-3*(B, S) be the result of deleting all clauses of B which contain an atom outside S. Then B is S_3-satisfiable if and only if *simplify-3*(B, S) is classically satisfiable.

Schaerf and Cadoli then obtain the following results for approximate inference:

THEOREM 2.5 ([SC95])
There exists an algorithm for deciding if $B \models^3_S \alpha$ and deciding $B \models^1_S \alpha$ which runs in $O(|B|.|\alpha|.2^{|S|})$ time.

The result above depends on a polynomial time satisfiability algorithm for belief bases and formulas in clausal form alone. This result has been extended in [CS95] for formulas in negation normal form, but is not extendable to formulas in arbitrary forms [CS96].

The good point of Schaerf and Cadoli's systems is that they present an incremental algorithm to test for S_i entailment as new elements are added to S. But there are two major limitations in their results:

1. The system is restricted to $\rightarrow$-free formulas and in negation normal form. In [CPW00] it is noted that the standard translation of formulas into clausal form does not preserve truth-values under the non-standard semantics of S_3.
2. The set S must be guessed at each step of the approximation; no method is given for the atoms to be added to S. Some heuristics for a specific application are presented in [tTvH97], but nothing is said about the general case.

In [FW01], we have extended S_3 to full propositional logic, given a sound and complete proof method for the extended system and showed how this proof method helps us to choose the atoms to add to S. In this paper, we will extend S_3 semantics further, compare it to da Costa's system C_1 and show what each system can offer to the other one. We provide an axiomatization for the extended version of S_3 based on the axiomatization of C_1 and provide a better proof method for C_1 than what was proposed before.

3 The Family of Logics $FineS_3$

In this section, we describe the logic S_3 extended to full propositional logic as originally in [FW01]. In fact, we are going to define a slightly distinct logic, that allows us to a better comparison with da Costa's C_1. Such a logic allows us to approximate classical logic in finer steps, and therefore we call it $FineS_3$.

The difference between S_3 and $FineS_3$ is the following. In [FW01], the set S which parameterized a logic in the family S_3 of logics consisted of a set of propositional letters, in which it followed Schaerf and Cadoli's initial proposal [SC95].

Here, however, we present a finer definition of S as a set of formulas closed under the formation of formulas. That is:

$$\alpha, \beta \in S \Longrightarrow \alpha \rightarrow \beta,\ \alpha \wedge \beta,\ \alpha \vee \beta,\ \neg\alpha \in S$$

In [FW01] we abused notation and stated that a formula $\alpha \in S$ whenever α's propositional were in S. This was equivalent to closing S under:

(a) formula construction; and

(b) subformula formation: If $\alpha \in S$ and β is a subformula of α, then $\beta \in S$.

Therefore, any logic in the original system is also a logic in $FineS_3$. Furthermore, we can now have $\alpha \in S$ without having any subformula of α in S, which was not possible in the original system S_3. As a result, here we can increase the set S in much finer steps, resulting in a much finer approximation of classical logic.

We present a binary semantics for $FineS_3$, which for that reason is called an $S_{3.2}$ semantics. It is remarkable that such semantic definition is exactly the same as that of [FW01], differing only in the notion of S in item (v) of Definition 3.1.

DEFINITION 3.1
A *propositional valuation* v_p is a function $v_p : \mathcal{P} \rightarrow \{0,1\}$. An $S_{3.2}$*-valuation* $v_S^{3.2}$ is a function, $v_S^{3.2} : \mathcal{L}_C \rightarrow \{0,1\}$, that extends a propositional valuation v_p (i.e., $v_S^{3.2}(p) = v_p(p)$), satisfying the following restrictions:

$$\begin{array}{lllll}
(i) & v_S^{3.2}(\alpha \wedge \beta) = 1 & \Leftrightarrow & v_S^{3.2}(\alpha) = v_S^{3.2}(\beta) = 1 \\
(ii) & v_S^{3.2}(\alpha \vee \beta) = 0 & \Leftrightarrow & v_S^{3.2}(\alpha) = v_S^{3.2}(\beta) = 0 \\
(iii) & v_S^{3.2}(\alpha \rightarrow \beta) = 0 & \Leftrightarrow & v_S^{3.2}(\alpha) = 1 \text{ and } v_S^{3.2}(\beta) = 0 \\
(iv) & v_S^{3.2}(\neg\alpha) = 0 & \Rightarrow & v_S^{3.2}(\alpha) = 1 \\
(v) & v_S^{3.2}(\neg\alpha) = 1, \alpha \in S & \Rightarrow & v_S^{3.2}(\alpha) = 0
\end{array}$$

Rules (i)–(iii) are exactly those of classical logic. Rules (iv) and (v) restrict the semantics of negation: rule (iv) states that if $v_S^{3.2}(\neg\alpha) = 0$, then negation behaves classically and forces $v_S^{3.2}(\alpha) = 1$; rule (v) states that if $v_S^{3.2}(\neg\alpha) = 1$, negation must behave classically only if $\alpha \in S$. Formulas outside S may behave classically or paraconsistently, i.e., both the formula and its negation may be assigned the truth value 1.

Note that an $S_{3.2}$-valuation is not uniquely defined by the propositional valuation it extends. This is due to the fact that if $\alpha \notin S$ and $v_S^{3.2}(\alpha) = 1$, the value of $v_S^{3.2}(\neg\alpha)$ can be either 0 (in which case α has a classical behavior) or 1 (in which case α behaves paraconsistently).

LEMMA 3.2
The valuation $v_S^{3.2}$ is determined by its value on the set $\mathcal{P} \cup \{\neg\alpha | \alpha \notin S\}$.

We define a formula α to be S*-valid* in $S_{3.2}$ if $v_S^{3.2}(\alpha) = 1$ for any $S_{3.2}$-valuation. A formula is S*-satisfiable* in $S_{3.2}$ if there is one $v_S^{3.2}$ such that $v_S^{3.2}(\alpha) = 1$. The $S_{3.2}$*-entailment* relationship between a set of formulas Γ and a formula α is represented as

$$\Gamma \models_S^{3.2} \alpha$$

and holds if every valuation $v_S^{3.2}$ that simultaneously satisfies all formulas in Γ also satisfies α. A formula is S-valid if it is entailed by $\emptyset$, represented as $\models_S^{3.2} \alpha$.

Lemma 3.2 suggests a translation between a formula in $FineS_3$ and one in classical logic, such that every formula of the form $\neg\alpha$ with $\alpha \notin S$ is mapped into a new propositional symbol $p_{\neg\alpha}$. Let α^{*S} be the translation of α, defined as:

$$\begin{array}{rcl} p^{*S} & = & p \\ (\alpha \circ \beta)^{*S} & = & \alpha^{*S} \circ \beta^{*S}, \quad \circ \in \{\wedge, \vee, \rightarrow\} \\ (\neg\alpha)^{*S} & = & \begin{cases} \neg(\alpha^{*S}), & \alpha \in S \\ p_{\neg\alpha} \quad , & \alpha \notin S \end{cases} \end{array}$$

The following properties of $FineS_3$ follow from this fact.

LEMMA 3.3
A formula α is S-satisfiable in $S_{3.2}$ iff $(\neg\alpha)^{*S}$ is classically satisfiable

LEMMA 3.4
Every formula $\alpha \notin S$ is satisfiable in $S_{3.2}$.

LEMMA 3.5
$S_{3.2}$-validity is subclassical. That is, any S-valid formula in $S_{3.2}$ is classically valid, for any S.

The deduction theorem holds for the $S_{3.2}$ semantics, as can be seen directly from the definitions.

LEMMA 3.6 (Deduction Theorem)
Let Γ be a finite set of formulas. Then

$$\Gamma \models_S^{3.2} \alpha \textit{ iff } \models_S^{3.2} \bigwedge \Gamma \rightarrow \alpha.$$

Now we examine a few examples of $S_{3.2}$ entailment.

EXAMPLE 3.7
Consider the formula $\alpha \vee \neg\alpha$. We show that it is a valid formula in $S_{3.2}$ for any S.

Indeed, if $\alpha \in S$, we are in a classical setting, so any valuation makes $\alpha \vee \neg\alpha$ true.

If $\alpha \notin S$, let $v_S^{3.2}$ be a valuation. If $v_S^{3.2}(\neg\alpha) = 1$, then $\alpha \vee \neg\alpha$ clearly is true. If, however, $v_S^{3.2}(\neg\alpha) = 0$, by rule (iv) above $v_S^{3.2}(\alpha) = 1$, so $\alpha \vee \neg\alpha$ is also true.

EXAMPLE 3.8
We now show that the $S_{3.2}$ semantics is paraconsistent. For that, consider the two propositions p and q and suppose that $p \notin S$; take a valuation $v_S^{3.2}$ such that $v_S^{3.2}(p) = 1$ and $v_S^{3.2}(q) = 0$, and consider the formula $(p \wedge \neg p) \rightarrow q$. By Lemma 3.2, the value of $v_S^{3.2}$ is not fully determined, so we fix $v_S^{3.2}(\neg p) = 1$. It is simple to verify now that the valuation thus constructed is such that $v_S^{3.2}((p \wedge \neg p) \rightarrow q) = 0$, that is the logic does not trivialize in the presence of inconsistency.

EXAMPLE 3.9
We now analyze the validity of Modus Ponens in $S_{3.2}$. The usual formulation of Modus Ponens, $\alpha \rightarrow \beta, \alpha \models_S^{3.2} \beta$, is valid in $S_{3.2}$; indeed, if $v_S^{3.2}$ satisfies α, the only possible way that it also satisfies $\alpha \rightarrow \beta$ is that it satisfies β, thus proving the entailment. Note that since no $\neg$-formula was involved, the reasoning is totally classical.

However, if we consider the version of Modus Ponens consisting of the translation of $\alpha \rightarrow \beta$ into $\neg\alpha \vee \beta$ (the only possible version of Modus Ponens in [SC95]), the situation changes completely if $\alpha \notin S$, for then we can have a valuation that satisfies both α and $\neg\alpha$ (and thus $\neg\alpha \vee \beta$), but that falsifies β, so that $\neg\alpha \vee \beta, \alpha \not\models_S^{3.2} \beta$.

Positive Axioms:

$(\rightarrow_1)$ $\alpha \rightarrow (\beta \rightarrow \alpha)$
$(\rightarrow_2)$ $(\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow (\alpha \rightarrow \gamma)$
$(\rightarrow_3)$ $\alpha, \alpha \rightarrow \beta / \beta$
$(\wedge_1)$ $\alpha \wedge \beta \rightarrow \alpha$
$(\wedge_2)$ $\alpha \wedge \beta \rightarrow \beta$
$(\wedge_3)$ $\alpha \rightarrow (\beta \rightarrow \alpha \wedge \beta)$
$(\vee_1)$ $\alpha \rightarrow \alpha \vee \beta$
$(\vee_2)$ $\beta \rightarrow \alpha \vee \beta$
$(\vee_3)$ $(\alpha \rightarrow \gamma) \rightarrow ((\beta \rightarrow \gamma) \rightarrow (\alpha \vee \beta \rightarrow \gamma))$

Negation Axioms:

$\mathbf{(neg_1)}$ $(\alpha \rightarrow \beta) \rightarrow ((\alpha \rightarrow \neg\beta) \rightarrow \neg\alpha)$, provided $\beta \in S$
$\mathbf{(neg_2)}$ $(\alpha \wedge \neg\alpha) \rightarrow \beta$, provided $\alpha \in S$
$\mathbf{(neg_3)}$ $\alpha \vee \neg\alpha$

FIG. 1. An Axiomatization for $FineS_3$

3.1 An Axiomatization for $FineS_3$

Consider the axioms in Figure 1. The *Positive Axioms* are precisely the classical axioms for the connectives $\rightarrow$, $\wedge$ and $\vee$; the rule of Modus Ponens is presented as $(\rightarrow_3)$. Our system differs from classical logic in the *Negation Axioms*.

In fact, without the proviso in axioms (neg_1) and (neg_2), the negation axioms are precisely the classical $\neg$-axioms. The $\neg$-introduction of axiom (neg_1) is restricted to the consequences of α in S. The trivialization of axiom (neg_2) is restricted to the members of S, that is, non-members of S can behave paraconsistently. Axiom (neg_3) tells us that the excluded middle is accepted unconditionally in our logic.

THEOREM 3.10
The axiomatization of Figure 1 is sound with respect to the $S_{3.2}$ semantics. That is, all formulas inferred from the $FineS_3$ axiomatization are true in all $S_{3.2}$-valuations.

PROOF. By a straightforward, but tedious, verification of the validity of the axioms. The positive axioms are dealt with by the classical part of the $S_{3.2}$ semantics. Axiom (neg_3) is dealt with in Example 3.8, and Modus Ponens is dealt with in Example 3.9. The validity of axioms (neg_1) and (neg_2) are easily verified. ∎

A formula α is *S-inconsistent* if it is provable that $\alpha \rightarrow (\beta \wedge \neg\beta)$ for some $\beta \in S$. Similarly, a set X of formulas is *S-inconsistent* if there are formulas $\chi_1, \ldots, \chi_n \in X$ such that $\bigwedge \chi_i \rightarrow (\beta \wedge \neg\beta)$ for some $\beta \in S$. A formula or set is *S-consistent* if it is not S-inconsistent.

The axiomatization above is *S-complete* iff for any S-consistent formula α there is an $S_{3.2}$ valuation $v_S^{3.2}$ such that $v_S^{3.2}(\alpha) = 1$. The proof of completeness follows a Lindenbaum construction.

A *maximal S-consistent set* (MSCS) is a set of formulas X such that:

- X is S-consistent; and
- there is no MSCS $X' \supset X$.

LEMMA 3.11
For every S-consistent formula α there is a MSCS X such that $\alpha \in X$.

PROOF. Consider an enumeration of the formulas $\beta_0, \beta_1, \ldots, \beta_i, \ldots$. Construct a sequence of sets X_i such that

$$\begin{array}{lcl} X_0 & = & \alpha \\ X_{i+1} & = & \begin{cases} X_i \cup \{\beta_i\}, & \text{if } X_i \cup \{\beta_i\} \text{ is } S\text{-consistent} \\ X_i, & \text{otherwise} \end{cases} \end{array}$$

Let $X = \bigcup_{i=0}^{\infty} X_i$. We claim that X is a MSCS, for the following reasons:

- X is S-consistent. Otherwise let X_{k+1} be the first element in the sequence that is S-inconsistent. Then we must have $X_{k+1} \neq X_k$, so $X_{k+1} = X_k \cup \{\beta_k\}$, but this is only possible if $X_k \cup \{\beta_k\}$ is S-consistent, contradicting the fact that X_{k+1} is S-inconsistent.
- X is maximal. Otherwise there is a MSCS $X' \supset X$. In this case there is a $\beta_k \in X' - X$. Consider the set $X_k \cup \{\beta_k\}$; if it is S-consistent, then $X_k \cup \{\beta_k\} = X_{k+1} \supset X$, which contradicts $\beta_k \notin X$. If $X_k \cup \{\beta_k\}$ is S-inconsistent, then X' is inconsistent because $X_k \supset X \supset X'$ and $\beta_k \in X'$, which contradicts the consistency of X'. Hence X must be maximal.

It is obvious that $\alpha \in X$, so the proof is finished. ∎

LEMMA 3.12
Let X be a MSCS. Let $v : \mathcal{L}_C \longrightarrow \{0,1\}$ such that $v(\alpha) = 1$ iff $\alpha \in X$. Then v is a $S_{3.2}$ valuation.

PROOF. We have to show that v satisfies the restrictions of Definition 3.1. Conditions $(i) - -(iii)$ are classical semantical conditions and are accounted for by the positive axioms, which are the classical ones.

We then show that: $(*) \quad v(\alpha) = 0 \Longrightarrow v(\neg\alpha) = 1$. For that, suppose $v(\alpha) = 0$. Then $\alpha \notin X$. Due to the maximality of X, there must be $\chi_1, \ldots, \chi_n \in X$ such that it is provable that $((\bigwedge \chi_i) \wedge \alpha) \rightarrow (\beta \wedge \neg\beta)$ for some $\beta \in S$. Therefore:

$$\begin{array}{l} \vdash ((\bigwedge \chi_i) \wedge \alpha) \rightarrow \beta \Longrightarrow \vdash (\bigwedge \chi_i) \rightarrow (\alpha \rightarrow \beta) \Longrightarrow \alpha \rightarrow \beta \in X \\ \vdash ((\bigwedge \chi_i) \wedge \alpha) \rightarrow \neg\beta \Longrightarrow \vdash (\bigwedge \chi_i) \rightarrow (\alpha \rightarrow \neg\beta) \Longrightarrow \alpha \rightarrow \neg\beta \in X \end{array}$$

It follows that both $\alpha \rightarrow \beta \in X$ and $\alpha \rightarrow \neg\beta \in X$. From Axiom (neg_1), it follows that $\neg\alpha \in X$, so $v(\neg\alpha) = 1$, proving $(*)$.

From (*) it follows that v obeys property (iv) of Definition 3.1. Indeed, suppose that $v(\neg\alpha) = 0$; if $v(\alpha) = 0$, then by $(*)$ we have that $v(\neg\alpha) = 1$, a contradiction. So we must have $v(\alpha) = 1$, satisfying (iv).

We then show that: $(**) \quad v(\alpha) = 1, \alpha \in S \Longrightarrow v(\neg\alpha) = 0$. For that, suppose $v(\alpha) = 1$ and $\alpha \in S$. Then $\alpha \in X$. Suppose now, for contradiction that $\neg\alpha \in X$. From Axiom (neg_2) we have that any $\beta \in X$, contradicting the consistency of X. So $\neg\alpha \notin X$ and $v(\neg\alpha) = 0$, thus proving $(**)$.

From $(**)$ we see that v obeys property (v) of Definition 3.1. Indeed, suppose that $v(\neg\alpha) = 1$ and $\alpha \in S$; if $v(\alpha) = 1$, by $(**)$ we have that $v(\neg\alpha) = 0$, a contradiction. So we must have $v(\alpha) = 0$, satisfying (v). ∎

THEOREM 3.13
The axiomatization of Figure 1 is complete with respect to the $S_{3.2}$ semantics.

PROOF. By Lemma 3.11 there is a MSCS X with $\alpha \in X$. Then, by Lemma 3.12, there is a $S_{3.2}$-valuation $v_S^{3.2}$ such that $v_S^{3.2}(\beta) = 1$ iff $\beta \in X$. In particular, $v_S^{3.2}(\alpha) = 1$. ∎

4 Relating $FineS_3$ to Da Costa's C_1

In this section, we introduce Da Costa's calculus C_1 by means of axioms and valuation semantics. We then show how our system relates to this calculus.

DEFINITION 4.1
A formula α is said to be *well behaved* if the principle of non-contradiction holds for α, i.e., if $\neg(\alpha \wedge \neg\alpha)$ holds. We use α^o to denote $\neg(\alpha \wedge \neg\alpha)$.

Da Costa's calculus [dC63] was introduced in order to deal with possible inconsistencies that should not damage reasoning by trivializing it. The idea is to block derivations from formulas which are not well behaved, isolating inconsistencies.

The following is an axiomatization of C_1:

$(\rightarrow_1)$ $\alpha \rightarrow (\beta \rightarrow \alpha)$
$(\rightarrow_2)$ $(\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow (\alpha \rightarrow \gamma)$
$(\rightarrow_3)$ $\alpha, \alpha \rightarrow \beta / \beta$
$(\wedge_1)$ $\alpha \wedge \beta \rightarrow \alpha$
$(\wedge_2)$ $\alpha \wedge \beta \rightarrow \beta$
$(\wedge_3)$ $\alpha \rightarrow (\beta \rightarrow \alpha \wedge \beta)$
$(\vee_1)$ $\alpha \rightarrow \alpha \vee \beta$
$(\vee_2)$ $\beta \rightarrow \alpha \vee \beta$
$(\vee_3)$ $(\alpha \rightarrow \gamma) \rightarrow ((\beta \rightarrow \gamma) \rightarrow (\alpha \vee \beta \rightarrow \gamma))$
$(\neg_1)$ $\beta^o \rightarrow (\alpha \rightarrow \beta) \rightarrow ((\alpha \rightarrow \neg\beta) \rightarrow \neg\alpha)$
$(\neg_2)$ $\alpha^o \wedge \beta^o \rightarrow ((\alpha \rightarrow \beta)^o \wedge (\alpha \wedge \beta)^o \wedge (\alpha \vee \beta)^o)$
$(\neg_3)$ $\alpha \vee \neg\alpha$
$(\neg_4)$ $\neg\neg\alpha \rightarrow \alpha$

A semantic for this system was given in [dCA77].

DEFINITION 4.2
An *$\mathcal{N}$-valuation* $v_{\mathcal{N}}$ is a function, $v_{\mathcal{N}} : \mathcal{L}_C \rightarrow \{0,1\}$, that extends a propositional valuation v_p (i.e., $v_{\mathcal{N}}(p) = v_p(p)$), satisfying the following restrictions:

(i)	$v_{\mathcal{N}}(\alpha \wedge \beta) = 1$	$\Leftrightarrow$	$v_{\mathcal{N}}(\alpha) = v_{\mathcal{N}}(\beta) = 1$
(ii)	$v_{\mathcal{N}}(\alpha \vee \beta) = 0$	$\Leftrightarrow$	$v_{\mathcal{N}}(\alpha) = v_{\mathcal{N}}(\beta) = 0$
(iii)	$v_{\mathcal{N}}(\alpha \rightarrow \beta) = 0$	$\Leftrightarrow$	$v_{\mathcal{N}}(\alpha) = 1$ and $v_{\mathcal{N}}(\beta) = 0$
(iv)	$v_{\mathcal{N}}(\alpha) = 0$	$\Rightarrow$	$v_{\mathcal{N}}(\neg\alpha) = 1$
(v)	$v_{\mathcal{N}}(\neg\neg\alpha) = 1$	$\Rightarrow$	$v_{\mathcal{N}}(\alpha) = 1$
(vi)	$v_{\mathcal{N}}(\beta^o) = v_{\mathcal{N}}(\alpha \rightarrow \beta) = v_{\mathcal{N}}(\alpha \rightarrow \neg\beta) = 1$	$\Rightarrow$	$v_{\mathcal{N}}(\alpha) = 0$
(vii)	$v_{\mathcal{N}}(\alpha^o) = v_{\mathcal{N}}(\beta^o) = 1$	$\Rightarrow$	$v_{\mathcal{N}}((\alpha * \beta)^o) = 1, * \in \{\rightarrow, \wedge, \vee\}$

An $\mathcal{N}$-valuation has the following properties:

LEMMA 4.3
(1) $v_{\mathcal{N}}(\alpha) = 1 \Leftrightarrow v_{\mathcal{N}}(\neg\alpha \wedge \alpha^o) = 0$
(2) $v_{\mathcal{N}}(\alpha^o) = 0 \Leftrightarrow v_{\mathcal{N}}(\alpha) = v_{\mathcal{N}}(\neg\alpha)$

It is not hard to see that the systems are very similar, although not equivalent. There are two main differences. First, the set of formulas for which $v_{\mathcal{N}}(\alpha^o) = 1$ does not correspond exactly to S:

LEMMA 4.4
If $\alpha \in S$, then $v_{\mathcal{N}}(\alpha^o) = 1$. The converse does not always hold, i.e., we may have an $\mathcal{N}$-valuation v such that $v(\alpha^o) = 1$ and $\alpha \notin S$.

PROOF. Follows directly from the definitions. $\alpha \in S$ means that α and $\neg\alpha$ must have opposite truth values, but $\alpha \notin S$ does not necessarily mean that α and $\neg\alpha$ have the same truth values. ∎

The second difference between the two systems is in the behavior of double negation. In C_1, double negation may be eliminated (but not introduced), while in S_3, α does not follow from $\neg\neg\alpha$.

Since we have sound and complete semantics for both systems, we can show how the systems relate to each other by checking the semantics.

PROPOSITION 4.5
Every $\mathcal{N}$-valuation is an $S_{3.2}$-valuation.

PROOF. Conditions (i),(ii),(iii), and (iv) are the same. We only have to check whether $v(\neg\alpha) = 1, \alpha \in S \Rightarrow v(\alpha) = 0$ follows from the definition of $\mathcal{N}$-valuation. We know that $\alpha \in S$ implies $v(\alpha^o) = 1$. And from $v(\neg\alpha) = v(\alpha^o) = 1$ it follows by part (2) of Lemma 4.3 that $v(\alpha) = 0$. ∎

PROPOSITION 4.6
Not every $S_{3.2}$-valuation is an $\mathcal{N}$-valuation.

PROOF. It suffices to see that we may have an $S_{3.2}$-valuation v so that for some formula α such that $\alpha \notin S$, we have $v(\neg\neg\alpha) = v(\neg\alpha) = 1$ and $v(\alpha) = 0$. This valuation fails to satisfy item (v) of the definition of $\mathcal{N}$-valuation. ∎

The two propositions show that *Fine*S_3 is actually more general than C_1.

COROLLARY 4.7
Let S be a fixed set of formulas closed under formula construction and let C_1^S be C_1 with an added axiom α^o for each $\alpha \in S$. Then Theorems(FineS_3) $\subseteq$ Theorems(C_1^3) $\subseteq$ Theorems($\mathcal{L}_C$).

We can adapt the tableaux system proposed in [FW01] for extended S_3 to build a proof method for C_1. In the rule where we had the proviso $\alpha \in S$ we now have α^o, and we have to add a rule to deal with double negation. The study of this tableaux system is left for future work.

5 Conclusion

In this paper, we have presented a new extended version of Cadoli and Schaerf's approximate entailment and given a sound and complete axiomatization for it. We have compared it to Da Costa's C_1 calculus for paraconsistent logic. We have shown that our system is more general than C_1, which can be seen both from the semantics as from the axiomatizations.

A proof method based on tableaux for paraconsistent logic has been proposed in [CLM92]. The idea behind their system is very similar to ours: the only difference from tableaux for classical logic is an extra condition for the rules involving negation. However, as in the calculus C_1 [dC63], the condition that a formula behaves classically is part of the language. As a result, the tableau construction sometimes loops. In our system, the fact that a formula is well behaved is not part of the language, but is stated in the meta-language, through the set S.

Future work includes studying the computational complexity of the tableaux system and extending Cadoli and Schaerf's S_1.

Acknowledgments

Marcelo Finger is partly supported by the Brazilian Research Council (CNPq), grant PQ 300597/95-5. Renata Wassermann was supported by FAPESP through grant 99/11602-6. This work was developed under the CNPq project APQ 468765/00-0.

References

[AB75] A.R Anderson and N.D Belnap. *Entailment: The Logic of Relevance and Necessity, Vol. 1*. Princeton University Press, 1975.

[CLM92] Walter A. Carnielli and Mamede Lima-Marques. Reasoning under inconsistent knowledge. *Journal of Applied Non-Classical Logics*, 2(1):49–79, 1992.

[CPW00] Samir Chopra, Rohit Parikh, and Renata Wassermann. Approximate belief revision. In *Proceedings of Workshop on Language, Logic and Information (WoLLIC)*, 2000.

[CS95] Marco Cadoli and Marco Schaerf. Approximate inference in default logic and circumscription. *Fundamenta Informaticae*, 23:123–143, 1995.

[CS96] Marco Cadoli and Marco Schaerf. The complexity of entailment in propositional multivalued logics. *Annals of Mathematics and Artificial Intelligence*, 18(1):29–50, 1996.

[dC63] Newton C.A. da Costa. Calculs propositionnels pour les systémes formels inconsistants. *Comptes Rendus d'Academie des Sciences de Paris*, 257, 1963.

[dCA77] Newton C. A. da Costa and Elias H. Alves. A semantical analysis of the calculi C_n. *Notre Dame Journal of Formal Logic*, 16, 1977.

[FW01] Marcelo Finger and Renata Wassermann. Tableaux for approximate reasoning. Technical report, Department of Computer Science - IME - University of São Paulo, 2001.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[Lev84] Hector Levesque. A logic of implicit and explicit belief. In *Proceedings of AAAI-84*, 1984.

[SC95] Marco Schaerf and Marco Cadoli. Tractable reasoning via approximation. *Artificial Intelligence*, 74(2):249–310, 1995.

[tTvH97] Annette ten Teije and Frank van Harmelen. Exploiting domain knowledge for approximate diagnosis. In M. Pollack, editor, *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 454–459, Nagoya, Japan, August 1997.