A Reasoning Method for a Paraconsistent Logic

Arthur Buchsbaum
Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente 225
22451-041 — Rio de Janeiro/RJ — Brazil
Email: arthur@inf.puc-rio.br

Tarcisio Pequeno
Laboratório de Inteligência Artificial
Universidade Federal do Ceará
Campus do Pici — Bloco 910
Caixa Postal 12.166
60021-970 — Fortaleza/CE — Brazil
Email: tarcisio@lia.ufc.br

June 17, 1992

Abstract

A proof method aiming to enable automation of reasoning in a paraconsistent logic, the calculus $\mathbf{C}_1^*$ of da Costa, is presented. The method is analytical, using a specially designed tableau system. Actually two tableau systems were created. One, with a small number of rules in order to be mathematically convenient, is used to prove the soundness and the completeness of the method. The other one, which is equivalent to the former, is a system of derived rules designed to enhance computational efficiency. A prototype based on this second system was effectively implemented.

1 Introduction

Our previous work on the problem of modelling and automatizing the reasoning required to produce a glimmer of intelligent behavior on machines has revealed the role that paraconsistent issues should play on it. This kind of reasoning is typically performed under conditions of incomplete and inaccurate knowledge, requiring inferences taken on basis of evidences that are only partial. Thus, *conservative*, deductive methods of reasoning will not suffice and the use of *creative*, superdeductive rules of inference is demanded. These rules, strictly speaking, are not generally sound, in the sense that situations might happen in which the premises of one rule are true, but its conclusion is not[1].

The adoption of this kind of rule makes the reasoning *nonmonotonic*: the addition of fresh information may eventually remove previous conclusions. This feature is so prominent that it has given its name to a whole area of investigation in Artificial Intelligence, and many systems of nonmonotonic logic have been designed so far.

On the other hand, there is another consequence of this lack of soundness which is frequently neglected in the literature on nonmonotonic logics: the incoming of contradictions. Unlike deduction, this kind of reasoning cannot be performed on local basis, without appealing to context. In the course of reasoning the arguments interfere with each other, generating conflicts and promoting the defeat of partial conclusions. Furthermore, there is no guarantee that every arising conflict can be resolved. It may perfectly happen two opposite partial conclusions having equal rights to be achieved or, even if there is not a perfect symmetry, it can happen anyway the available knowledge not enabling a clear decision in favor of one of the alternatives.

In Pequeno [7] it is suggested that, in order to give the right account for the situation and provide a logical analysis of the consequences of the evidences available, these contradictions should be assimilated in a single set of believes and reasoned out just as any other theory. This would require, of course, the employment of a *paraconsistent logic*, a logic able to perform sensible reasoning from theories comprising contradictions. In Pequeno & Buchsbaum [8] it is presented a paraconsistent logic specially designed for this purpose. It is called "logic of epistemic inconsistency", LEI for short.

The recognition of the import of paraconsistency for the automatization of reasoning has been our main motivation for the study of proof methods for paraconsistent logics suitable to computer implementation. We have worked out reasoning methods by tableau for an assortment of paraconsistent calculus, and even a generalized framework for tableaux has been designed for the job [3]. A proof method for LEI is presented in [4].

The reasoning method presented here is a tableau system, sound and complete for $\mathbf{C}_1^*$ (see da Costa [5]). Considering that paraconsistent logic

[1]An example of this kind of rule is *default reasoning*, a style of reasoning in which a conclusion is taken, when authorized by a "default rule", if its negation cannot be inferred (see Reiter [9]).

deviates from classical logic essentially in the way they see negation, we decided to emphasize here the tableau system relative to the connective structure of the logic. As a matter of fact, two tableau systems were developed. In section 3 the system $\mathbf{SC}_1^*$, which is economic in rules, is presented along with the sketches for the proofs of its soundness and completeness with respect to the semantics of $\mathbf{C}_1^*$. It was devised in the sake of mathematical convenience, being not suitable for implementation. In section 4 the system $\mathbf{S'C}_1^*$, designed to enable better implementation, is described. Comparing with $\mathbf{SC}_1^*$, $\mathbf{S'C}_1^*$ has more rules that are, generally, more specialized and thus prone to produce smaller trees than those resulting from the applications of the rules in $\mathbf{SC}_1^*$. The two systems are equivalent in the sense that they are able to prove the same theorems. A prototype has been worked out, based entirely in this second system.

2 Paraconsistent Logic

In a paraconsistent logic, otherwise classical logic, from the fact that A and $\neg A$ are deducible it does not follow that any formula B is deducible as well. Indeed, this property is one of the three properties stated by da Costa to be satisfied by a logic in order to be called paraconsistent. The other two da Costa's principles are:

- The noncontradiction law, which, according to da Costa, can be expressed by the fact that $\neg(A \wedge \neg A)$ is a logical theorem for any formula A in the language, should not be valid.
- All theorems of classical logic which do not interfere with the two properties mentioned before should be maintained.

The first two properties require the weakening of the classical axiomatics, while the third one states that this weakening should not be greater than it is really necessary.

The noncontradiction law in classical logic is supported, in many formulations, by the following axiom:

$$(A \to B) \to \big((A \to \neg B) \to \neg A\big).$$

The paraconsistency is attained in $\mathbf{C}_1^*$ exactly by restricting this axiom to those formulas B for which the noncontradiction property $\neg(B \wedge \neg B)$ is explicitly affirmed. Thus the axiom of the absurd in $\mathbf{C}_1^*$ becomes

$$\neg(B \wedge \neg B) \to \Big((A \to B) \to \big((A \to \neg B) \to \neg A\big)\Big).$$

The calculus $\mathbf{C}_1^*$ is constituted by taking the classical axioms (as in Kleene [6]), with the axiom of the absurd modified as above. Furthermore,

$$A \vee \neg A \qquad \text{and} \qquad \neg\neg A \rightarrow A$$

are added as axioms, since they are no longer theorems. Additional axioms, intended to preserve noncontradiction property under composition of formulas, are also provided:

$$A^\circ \wedge B^\circ \rightarrow (A \rightarrow B)^\circ,$$
$$A^\circ \wedge B^\circ \rightarrow (A \wedge B)^\circ,$$
$$A^\circ \wedge B^\circ \rightarrow (A \vee B)^\circ,$$
$$\forall x\, A^\circ \rightarrow (\forall x\, A)^\circ,$$
$$\forall x\, A^\circ \rightarrow (\exists x\, A)^\circ.$$

We adopted here an abbreviation that is quite usual in the literature on paraconsistent logic: A° for $\neg(A \wedge \neg A)$.

There is still another axiom, which is a theorem of classical quantificational logic, but in $\mathbf{C}_1^*$ it is independent to the other axioms just given:

$$A \leftrightarrow A',$$

where A and A' stand for identical formulas, except for congruence and presence of vacuous quantifiers.

The calculus above meets da Costa's requirements. For instance, all the rules of natural deduction, with exception to the $\neg-$introduction rule, are valid. The calculus $\mathbf{C}_1^*$ can be regarded as giving an axiomatization for the negation that makes it weaker than the classical negation as much as needed to avoid trivialization in the presence of contradiction.

3 $\mathbf{SC}_1^*$: A Tableau System for $\mathbf{C}_1^*$

A tableau system is a method of proof by refutation which consists in a generation of a tree (tableau), originated from an *initial tableau.* This initial tableau is normally a node constituted by the negation of the theorem to be demonstrated (strong, classical negation in case of calculus $\mathbf{C}_1^*$). Step by step, after the choice of a node still not used up, the growing of the tree proceeds from all descendant leaves of this node, located in *open branches*, by the application of a *tableau expansion rule* (in general cases, an expansion rule depends not only on the formula in the leaf, but on its whole ascending branch). The proliferation of branches is sustained by a *closure operation*, by which a branch is closed according to the occurrence of a certain condition. The goal of the procedure is to *close* all branches, thereby proving the given theorem. Semantically, the closure of all branches of the tableau establishes the unsatisfiability of the formula in its root.

The tableau method is said to be analytical, since it proceeds by a gradual decomposition of the given formula, in opposition to conventional methods of automated proof by resolution. In the tableau method we deal directly with the given formulas, without appealing, for example, to normalization into clauses. In order to define a tableau system three elements must be provided: a collection of *expansion rules*, henceforward named simply *rules*, a *branch closure criterion*, and an *initialization function*, that produces an *initial tableau* from the formula to be verified.

In the presentation of our tableau system we shall use another abbreviation: $\sim A$ to $\neg A \wedge A^{\circ}$. The symbol "$\sim$" is named *strong negation* and it has all properties of classical negation. The abbreviations introduced so far, A° and $\sim A$, are so convenient that they were adopted even in the implemented version of the reasoning system, which is able to handle formulas that include them. The tableau system for $\mathbf{C}_1^*$ ($\mathbf{SC}_1^*$) is initialized with the *strong negation* of the proposed theorem. The closure criterium of branches for $\mathbf{SC}_1^*$ says that a branch is closed if it contains a complementary pair of formulas A and $\sim A'$, whereon A and A' are identical, except for congruence and presence of vacuous quantifiers.

The rules of $\mathbf{SC}_1^*$ follow below. The name of each rule tells the kind of formulas to which it can be applied. The sign "#" stands for whatever one of the signs "$\rightarrow$", "$\wedge$" or "$\vee$", and the sign "Q" represents a quantifier ("$\forall$" or "$\exists$"). Concerning the rules for quantifiers, let $\mathcal{L}$ be the initial language of $\mathbf{SC}_1^*$ (where axioms and theorems to be proved from them can be formulated), α the formula from which the initial tableau considered is defined, $\mathcal{L}_\alpha$ the language with the same logical symbols of $\mathcal{L}$ whose non logical symbols are those of α, $\mathcal{L}'_\alpha$ the language obtained from $\mathcal{L}_\alpha$ by adding an infinite denumerable number of new constants, t the first closed term of $\mathcal{L}'_\alpha$ such that the appointed formula does not occur in the branch considered, and finally c the first constant of $\mathcal{L}'_\alpha$ which does not occur in the branch considered.

$$\begin{array}{ccc} \underline{\text{Rule } A \rightarrow B} \\ A \rightarrow B \\ \diagup \qquad \diagdown \\ \sim A \qquad\qquad B \end{array} \qquad \begin{array}{c} \underline{\text{Rule } A \wedge B} \\ A \wedge B \\ | \\ A \\ | \\ B \end{array} \qquad \begin{array}{ccc} \underline{\text{Rule } A \vee B} \\ A \vee B \\ \diagup \qquad \diagdown \\ A \qquad\qquad B \end{array}$$

$$\begin{array}{c}\underline{\text{Rule } \sim(A \to B)}\\ \sim(A \to B)\\ | \\ A \\ | \\ \sim B\end{array} \qquad \begin{array}{c}\underline{\text{Rule } \sim(A \land B)}\\ \sim(A \land B)\\ \swarrow \quad \searrow \\ \sim A \qquad \sim B\end{array} \qquad \begin{array}{c}\underline{\text{Rule } \sim(A \lor B)}\\ \sim(A \lor B)\\ | \\ \sim A \\ | \\ \sim B\end{array}$$

$$\begin{array}{c}\underline{\text{Rule } \neg(A\#B)}\\ \neg(A\#B)\\ \begin{array}{ccc} | & | & | \\ \sim(A\#B) & \sim A^\circ & \sim B^\circ\end{array}\end{array} \qquad \begin{array}{c}\underline{\text{Rule } \neg\neg A}\\ \neg\neg A\\ \swarrow \quad \searrow \\ \sim\neg A \qquad \sim A^\circ\end{array} \qquad \begin{array}{c}\underline{\text{Rule } \sim\neg A}\\ \sim\neg A\\ | \\ A\end{array}$$

$$\begin{array}{c}\underline{\text{Rule } \forall x\, A}\\ \forall x\, A\\ | \\ A(x|t) \\ | \\ \forall x\, A\end{array} \qquad \begin{array}{c}\underline{\text{Rule } \exists x\, A}\\ \exists x\, A\\ | \\ A(x|c)\end{array} \qquad \begin{array}{c}\underline{\text{Rule } \sim\forall x\, A}\\ \sim\forall x\, A\\ | \\ \sim A(x|c)\end{array}$$

$$\begin{array}{c}\underline{\text{Rule } \sim\exists x\, A}\\ \sim\exists x\, A\\ | \\ \sim A(x|t) \\ | \\ \forall x \sim A\end{array} \qquad \begin{array}{c}\underline{\text{Rule } \neg Qx\, A}\\ \neg Qx\, A\\ \swarrow \quad \searrow \\ \sim Qx\, A \qquad \sim A^\circ(x|c)\end{array}$$

We call *satisfiable* a tableau having at least a *satisfiable branch* (a branch in which all the formulas attached to it are simultaneously satisfiable). About each one of the rules given above, it can be demonstrated that, if A is a satisfiable formula, then so is some branch of an application of a rule to A. It can be shown that, if T is a satisfiable tableau, the tableau resulting from the expansion of T by the application of one of the rules above is satisfiable as well. Therefore, if A is satisfiable, any tableau for A is satisfiable too. It follows that the existence of an unsatisfiable tableau for A implies the unsatisfiability of A. From the fact that a confutation (a tableau whose all branches are closed) in $\mathbf{SC}_1^*$ is unsatisfiable, the soundness of $\mathbf{SC}_1^*$ with respect to the semantics of $\mathbf{C}_1^*$ can be concluded: if there is a confutation for $\sim A$ in $\mathbf{SC}_1^*$, A is logically valid in $\mathbf{C}_1^*$ and thus a theorem of $\mathbf{C}_1^*$.

The argument to establish the completeness of $\mathbf{SC}_1^*$ runs along the following lines. We can show that if a developing complete sequence of tableaux obtained by a depth first search does not contain a confutation, then it contains a tableau with an *open exhausted branch* (an open branch whereon the only formulas not used are atomic, negations of atomic formulas, or strong negations of atomic formulas), or it tends to an infinite limit tree containing an infinite exhausted branch. As any open exhausted branch is satisfiable, we conclude that the formula in the root node is satisfiable. So, if A is unsatisfiable, then the developing complete sequence of tableaux obtained from the initial tableau for A and from a depth-first search must contain a confutation. $\mathbf{SC}_1^*$ is therefore complete, in the sense that for any formula A, if A is valid, and hence a theorem of $\mathbf{C}_1^*$, then there exists a confutation for $\sim A$.

Our method is a procedure to find this existing confutation. Detailed proofs can be found in Buchsbaum [1].

4 $\mathbf{S'C}_1^*$: A Tableau System for Implementation

In this section we present a tableau system derived from $\mathbf{SC}_1^*$, designed to improve efficiency of implementation. The basic idea is to provide $\mathbf{S'C}_1^*$ with a greater number of more specialized rules. The generated trees tend to contain less nodes then those from $\mathbf{SC}_1^*$. The closure criterium of $\mathbf{S'C}_1^*$ was also modified: a branch is closed if it contains either a complementary pair of formulas A and $\sim A'$, or any formula of one of the forms $\sim\big((A \wedge \neg A')^\circ\big)$, $\sim(A^{\circ^\circ})$, or $\sim\big((\sim A)^\circ\big)$, whereon A and A' are identical except for congruence. The new rules of $\mathbf{S'C}_1^*$ are given below.

$$\begin{array}{c} \underline{\text{Rule } \ A^\circ} \\ A^\circ \\ \swarrow \quad \searrow \\ \sim A \qquad \sim\neg A \end{array} \qquad \begin{array}{c} \underline{\text{Rule } \ \sim A^\circ} \\ \sim A^\circ \\ | \\ A \\ | \\ \neg A \end{array} \qquad \begin{array}{c} \underline{\text{Rule } \ \neg A^\circ} \\ \neg A^\circ \\ | \\ \sim A^\circ \end{array}$$

$$\begin{array}{c} \underline{\text{Rule } \ \sim\sim A} \\ \sim\sim A \\ | \\ A \end{array} \qquad \begin{array}{c} \underline{\text{Rule } \ \neg\sim A} \\ \neg\sim A \\ | \\ A \end{array} \qquad \begin{array}{c} \underline{\text{Rule } \ A \leftrightarrow B} \\ A \leftrightarrow B \\ \swarrow \quad \searrow \\ \begin{array}{cc} A & \sim A \\ | & | \\ B & \sim B \end{array} \end{array}$$

$$\begin{array}{ccc} \multicolumn{3}{c}{\underline{\text{Rule } \sim(A \leftrightarrow B)}} \\ & \sim(A \leftrightarrow B) & \\ A & & \sim A \\ \sim B & & B \end{array}$$

$$\begin{array}{ccc} \multicolumn{3}{c}{\underline{\text{Rule } \neg(A \leftrightarrow B)}} \\ & \neg(A \leftrightarrow B) & \\ \sim(A \leftrightarrow B) & \neg(A \rightarrow B) & \neg(B \rightarrow A) \end{array}$$

$\mathbf{S'C}_1^*$ is a conservative extension of $\mathbf{SC}_1^*$. Its additional rules can be regarded as *derived rules* from those in $\mathbf{SC}_1^*$. They can be shown equivalent in the sense that a formula can generate a confutation in $\mathbf{S'C}_1^*$ if and only if it can generate a confutation in $\mathbf{SC}_1^*$. Proofs, again, may be found in Buchsbaum [1].

5 Conclusions

At this point, further development can take two different lines. From one side, the general method could be explored on its applicability to other systems of non classical logic. Actually reasoning methods to several other paraconsistent and paracomplete logics have already been designed (see Buchsbaum & Pequeno [2]), and a general framework for non classical tableaux have been provided in Buchsbaum & Pequeno [3].

Another line of development would be the improvement of the prototype as a prover. The main function of the prototype, as it was constructed, is to demonstrate the feasibility of the method and to be useful as a laboratory tool for forthcoming work. Its architecture was designed in order to enhance the construction of different prototypes by changing the existing rules or including new rules. In order to become a practical useful reasoning tool, it should be equipped with suitable strategies to improve efficiency and an adequate interface to make it more friendly.

References

[1] Arthur Buchsbaum. Um método automático de prova para a lógica paraconsistente. Master's thesis, Pontifícia Universidade Católica do Rio de Janeiro, 1988.

[2] Arthur Buchsbaum and Tarcisio Pequeno. Raciocínio automático em lógicas paraconsistentes e/ou paracompletas. In *6º Simpósio Brasileiro em Inteligência Artificial*, pages 1–15, 1989.

[3] Arthur Buchsbaum and Tarcisio Pequeno. O método dos tableaux generalizado e sua aplicação ao raciocínio automático em lógicas não clássicas. *O que nos faz pensar — Cadernos do Departamento de Filosofia da PUC-Rio*, 3:81–96, September 1990.

[4] Marcelo Correa, Arthur Buchsbaum, and Tarcisio Pequeno. Raciocínio automático com conhecimento incompleto e inconsistente I: Um sistema de tableaux para LEI. In *Proceedings of the 9th Brazilian Symposium of Artificial Intelligence*, pages 281–296, 1992.

[5] Newton C. A. da Costa. On the theory of inconsistent formal systems. *Notre Dame Journal of Formal Logic*, 15:497–510, 1974.

[6] Stephen Cole Kleene. *Introduction to Metamathematics.* Wolters-Noordhoff, North Holland and American Elsevier, 1974.

[7] Tarcisio Pequeno. A logic for inconsistent nonmonotonic reasoning. Technical Report 6, Department of Computing, Imperial College, London, 1990.

[8] Tarcisio Pequeno and Arthur Buchsbaum. The logic of epistemic inconsistency. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, pages 453–460. Morgan Kaufmann, 1991.

[9] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.