USING LABELS IN A PARACONSISTENT AND NONMONOTONIC SEQUENT CALCULUS

Ana Teresa de Castro Martins† Ruy J. G. B. de Queiroz‡
Departamento de Informática
Universidade Federal de Pernambuco
C.P. 7851, Recife, PE 50732-970 Brasil
e-mail: atcm@di.ufpe.br, ruy@di.ufpe.br

Abstract

The aim of this paper is to present a labelled sequent calculus to a nonmonotonic and paraconsistent Logic of Evidence able to tolerate contradictions over opposite evidences without proving anything. Evidences are ?-marked and they must be previously ordered. The role of the labels are stressed as a tool to handle meta level features side-by-side to object level ones.

Key Words: Nonmonotonic Reasoning, Paraconsistent Logic, Labelled Deductive System.

1 Introduction

The purpose of this paper is to show the role of the labels in a sequent calculus to a nonmonotonic and paraconsistent Logic of Evidences. We will assume some previous knowledge of the labelling approach to deductive systems (see [Gabbay 9?]) and the sequent style for classical inference rules (see [Gentzen 35]).

Our Logic of Evidence can be used to support the reasoning of an intelligent agent. Generally speaking, an intelligent agent must be able to take decisions based on incomplete and imprecise informations eventually involving arguments that interfere to each other and possibly causing contradictions. A typical example can be found in a diagnostic medical problem. A doctor should indicate the best treatment for a disease even if he does not have the means of analysing all necessary symptoms, or even if there is no consensus among specialists of what the best treatment is to consider.

How can this form of reasoning be handled properly? In Automated Reasoning area we can mention a formalism that aim to model that knowledge properly: the nonmonotonic logic (see [Reiter 80]). This logic rejects the monotonicity property of classical logic, namely the property that if a conclusion is warranted on the basis of certain premises, no additional information can retract it. In everyday life it does not make sense, since it is impossible to conclude anything for sure based on uncertain knowledge. The most you can do in this situation is to ignore exceptional cases for a while and reason as you would do in a hypothetical ideal situation. Unfortunately contradictions arise and must be treated appropriately.

The generation of contradictory conclusions is a crucial problem under the scope of nonmonotonic formalisms. A credulous approach that allows to handle contradictions in the same theory is the paraconsistent

---

*Research sponsored by PICD/CAPES
†On leave from Universidade Federal do Ceara
‡Research fellow, CNPq, Brasil
logic ([da Costa 74]). This logic relaxes the *reductio ad absurdum* rule of classical logic, i.e., contradiction is tolerate without trivializing the deductive system, without allowing to prove anything from that. Using this logic, you can retain the contradiction and delay the decision of choosing one of the opposite points of view until you have sufficient information to decide.

The complementary roles of nonmonotonic and paraconsistent reasonings are stressed in [Pequeno 90]. In this work, Pequeno presents an Inconsistent Default Logic (IDL), a nonmonotonic logic able to tolerate contradiction. This logic extends the classical logic language with *?-formulas* representing default (plausible) conclusions. The paraconsistent aspect of IDL is, by itself, another logic, the Logic of Epistemic Inconsistency (LEI) presented in [Pequeno & Buchsbaum 91].

In this work we will present a *labelled* sequent calculus to our Logic of Evidence. This logic was strongly influenced by IDL/LEI. It is also a nonmonotonic and paraconsistent logic with the same augmented language but with a somewhat different axiomatic base where the question mark '?' signalizes evidences to the marked formulas even if they are not plausible ones. The main difference is that the nonmonotonicity property is not obtained by a default rule. Instead, we will introduce a structure in this enhanced language, a partial order '<'. This order reflects a different *force* of *?-formulas*, a degree of belief in their evidences.

This work is divided in four sections. In the second section we will present the labelled rules to manipulate this logic. In the third section we will give a proof example to clarify the use of some rules and the role of the labels in our calculus. Finally we will conclude projecting some possible extensions to our work.

## 2 The *Labelled* Sequent Rules for the Logic of Evidence

In this section we will present our labelled rules for the Logic of Evidence. We will adopt a language $L^?$ for the Logic of Evidence, i.e., a classical logic language $L$ augmented with *?-formulas*. We will use capital roman letters to denote arbitrary *?-free* formulas and small roman letters to denote any formula in $L^?$. $C$, $C'$, $D$ and $D'$ represent a (possibly empty) list of *?-free* formulas and the dots '...' represent a (possible empty) list of *?-formulas*.

Attached to formulas we have labels. The pattern of a *labelled* formula is 'a : b' where 'a' belongs to $L^?$ and 'b' is its corresponding label. The structure of the labels and their role in the proof will be explained in the next section through a proof example.

The right and left *?-free* formulas in the sequent are simply set of *?-free* formulas. There are no kind of structure in this set. To add *?-formulas* in the calculus, it is necessary to introduce a structure between them: a partial order '<'. *?-Formulas* are ordered in crescent order from right to left in the left hand side of the sequent and from left to right in the opposite side. *?-Formulas* with the same order or not comparable are written one below the others in a vertical (imaginary) line in relation to $\vdash$. For example:

$$\frac{\frac{C \vdash a^? \quad e? \vdash f^?}{c? \vdash b^? \quad \text{\ldots}}}{C' \vdash D}$$

represents 'c? < a??', 'e? < b??', 'd?? < f??', 'e?? < f??', but 'a??' and 'b??' are examples of uncomparable formulas. Even though the sets $C$ and $D$ of *?-free* formulas contain no order, any formula in these sets is greater than any *?-formula* and, for this reason, they are written in the extremes of each side of the sequent.

We are not interested here in what kind of order we are talking about. It could be anyone: specificity, probability, plausibility, priority or any other. We are also not interested in how we should order *?-formulas* in the sequent to reflect this structure. These are all epistemologic questions and do not belong to our scope of investigation. We are just concerned in how can we reason with uncertain and previously ordered points of view without trivializing the reasoning — without proving anything — when some of these points of view are contradictory.
The sequent calculus is divided into three groups of rules as usual: Structural, Identity and Operational rules. All classical rules are allowed for "?-free formulas with the exception of Left Weakening to preserve the nonmonotonicity property of our logic. These classical rules will be assumed to simplify our presentation. The inference rules are:

**Structural Rules**

\[
\begin{align*}
& \text{LX} & \frac{C \quad x : a \quad \vdash \quad \Gamma}{C \quad y : b \quad \vdash \quad \Gamma} & \quad \frac{C \quad \vdash \quad \Gamma}{C \quad y : b \quad \vdash \quad \Gamma} & \quad \text{RX} \\
& \text{LC} & \frac{C \quad y : a \quad \vdash \quad \Gamma}{C \quad x : a \quad \vdash \quad \Gamma} & \quad \frac{C \quad \vdash \quad \Gamma}{C \quad y : a \quad \vdash \quad \Gamma} & \quad \text{RC}
\end{align*}
\]

provided 'x' and 'y' are reducible to the same normal form.

\[
\begin{align*}
& \text{LA} & \frac{C \quad x : b \quad \vdash \quad \Gamma}{C \quad x : b \quad \vdash \quad \Gamma} & \quad \frac{C \quad \vdash \quad \Gamma}{C \quad x : a? \quad \vdash \quad \Gamma} & \quad \text{RA} \\
& \text{LA} & \frac{C \quad \vdash \quad \Gamma}{C \quad x : a? \quad \vdash \quad \Gamma} & \quad \frac{C \quad \vdash \quad \Gamma}{C \quad x : a \quad \vdash \quad \Gamma} & \quad \text{RA}
\end{align*}
\]

provided 'b' is 'a' or 'a?'.

**Identity Rules**

\[
x : a \vdash y : a \quad \text{Identity}
\]

provided 'x' and 'y' are reducible to the same normal form.

\[
\frac{C \quad \vdash \quad \Gamma}{C \quad \vdash \quad \Gamma} \quad \frac{C \quad \vdash \quad \Gamma}{C \quad \vdash \quad \Gamma} \quad \frac{C \quad \vdash \quad \Gamma}{C \quad \vdash \quad \Gamma}
\]

provided 'f' and 'f' are reducible to the same normal form.

\[
\frac{C \quad \vdash \quad \Gamma}{C \quad \vdash \quad \Gamma} \quad \frac{C \quad \vdash \quad \Gamma}{C \quad \vdash \quad \Gamma}
\]

provided 'f' and 'f' are reducible to normal forms 'p' and 'q' respectively where 'p = c(q)'.

**Operational Rules**

\[
\begin{align*}
& \text{L} \rightarrow & \frac{\Gamma \vdash \text{mod}_1(x) : a \quad \vdash \quad \Gamma}{\Gamma \vdash x : \neg a \quad \vdash \quad \Gamma} & \quad \frac{\Gamma \vdash \text{mod}_2(x) : a \quad \vdash \quad \Gamma}{\Gamma \vdash x : \neg a \quad \vdash \quad \Gamma} & \quad \text{R} \rightarrow \\
& \text{Ll} \wedge & \frac{\Gamma \vdash f s t_1(x) : a \quad \vdash \quad \Gamma}{\Gamma \vdash x : (a \wedge b) \quad \vdash \quad \Gamma} & \quad \frac{\Gamma \vdash s n d_1(x) : b \quad \vdash \quad \Gamma}{\Gamma \vdash x : (a \wedge b) \quad \vdash \quad \Gamma} & \quad \text{L2} \wedge \\
& & \frac{\Gamma \vdash f s t_2(x) : a \quad \vdash \quad \Gamma}{\Gamma \vdash \Gamma \quad \vdash \quad \Gamma} & \quad \frac{\Gamma \vdash s n d_2(x) : b \quad \vdash \quad \Gamma}{\Gamma \vdash \Gamma \quad \vdash \quad \Gamma} & \quad \text{R} \wedge \\
& & \frac{\Gamma \vdash \Gamma \quad \vdash \quad \Gamma}{\Gamma \vdash \Gamma \quad \vdash \quad \Gamma}
\end{align*}
\]
\[
\frac{C \ldots l_1(x) : a \ldots \vdash \ldots D}{C, C' : a \lor b \ldots \vdash \ldots D, D'} \quad \text{LV}
\]

\[
\frac{C \ldots \vdash \ldots b(x) : a \ldots D}{C \ldots \vdash \ldots x : (a \lor b) \ldots \vdash \ldots D, D'} \quad \text{RIV}
\]

\[
\frac{C \ldots \vdash \ldots r_2(x) : b \ldots D}{C \ldots \vdash \ldots x : (a \lor b) \ldots \vdash \ldots D} \quad \text{R2V}
\]

\[
\frac{C \ldots \vdash \ldots \text{var}_1(x) : a \ldots D}{C, C' : a \rightarrow b \ldots \vdash \ldots D, D'} \quad \text{L-}
\]

\[
\frac{C \ldots \vdash \ldots \text{app}_1(x, \text{var}_1(x)) : b \ldots D}{C, C' \ldots \vdash \ldots x : (a \rightarrow b) \ldots \vdash \ldots D, D'}
\]

\[
\frac{C \ldots \vdash \ldots \text{var}_2(x) : a \ldots \vdash \ldots D}{C \ldots \vdash \ldots \text{app}_2(x, \text{var}_2(x)) : b \ldots \vdash \ldots D}
\]

\[
\frac{C \ldots \vdash \ldots \rho_1(x) : a \ldots \vdash \ldots D}{C \ldots \vdash \ldots x : a \ldots \vdash \ldots D, D'} \quad \text{R-}
\]

\[
\frac{C \ldots \vdash \ldots \rho_2(x) : a \ldots D}{C \ldots \vdash \ldots x : a \ldots \vdash \ldots D}
\]

\[
\frac{C \ldots \vdash \ldots \text{ld}_\varnothing (\Gamma(c(-x) + c(y)/c(\lambda x.y))) : ((\neg a) ? \lor b) \ldots \vdash \ldots D}{C \ldots \vdash \ldots \Gamma(c(\lambda x.y)) : (a \rightarrow b) \ldots \vdash \ldots D}
\]

where ‘\(\Gamma\)’ is a (possibly empty) list of destructor operators. This terminology is also valid for the rules below.

\[
\frac{C \ldots \vdash \ldots \text{rd}_\varnothing (\Gamma[c(-x) + c(y)/c(\lambda x.y)]) : ((\neg a) ? \lor b) \ldots \vdash \ldots D}{C \ldots \vdash \ldots \Gamma(c(\lambda x.y)) : (a \rightarrow b) \ldots \vdash \ldots D} \quad \text{RD}_-\]

\[
\frac{C \ldots \vdash \ldots \text{ld}_\varnothing (\Gamma[c(x) + c(y)/c(x + y)]) : ((\neg a) ? \lor b) \ldots \vdash \ldots D}{C \ldots \vdash \ldots \Gamma(c(\lambda x.y)) : (a \rightarrow b) \ldots \vdash \ldots D} \quad \text{RD}_\varnothing
\]

\[
\frac{C \ldots \vdash \ldots \text{rd}_\varnothing (\Gamma[c(x) + c(y)/c(x + y)]) : ((\neg a) ? \lor b) \ldots \vdash \ldots D}{C \ldots \vdash \ldots \Gamma(c(\lambda x.y)) : (a \rightarrow b) \ldots \vdash \ldots D}
\]

\[
\frac{C \ldots \vdash \ldots \text{ld}_\varnothing (\Gamma[c(x) + c(y)/c(x + y)]) : ((\neg a) ? \lor b) \ldots \vdash \ldots D}{C \ldots \vdash \ldots \Gamma(c(\lambda x.y)) : (a \rightarrow b) \ldots \vdash \ldots D}
\]

\[
\frac{C \ldots \vdash \ldots \text{ld}_\varnothing (\Gamma[c(x) + c(y)/c(x + y)]) : ((\neg a) ? \lor b) \ldots \vdash \ldots D}{C \ldots \vdash \ldots \Gamma(c(\lambda x.y)) : (a \rightarrow b) \ldots \vdash \ldots D}
\]

\[
\frac{C \ldots \vdash \ldots \text{ld}_\varnothing (\Gamma[c(x) + c(y)/c(x + y)]) : ((\neg a) ? \lor b) \ldots \vdash \ldots D}{C \ldots \vdash \ldots \Gamma(c(\lambda x.y)) : (a \rightarrow b) \ldots \vdash \ldots D}
\]

The Structural rules for \#-formulas must preserve the partial order between them. The Exchange rule permits moving the position of the formulas in the sequent. For our calculus, we can use the Exchange rule only for \#-formulas situated in the same vertical line (equivalent or not comparable formulas) because if you change the position of a \#-formula in the horizontal line you will be changing the order between them. The Contraction rule shows that two or more occurrences of the same formula can be mixed in one occurrence. It is also not possible to use Contraction for \#-formulas unless they are situated in the same vertical line, because two occurrences of the same \#-formula in a horizontal line have different order and they are not really the same formula.

We have introduced one more Structural rule: the Absorption rule. The LA rule permits a weaker ‘\(a\)’ or ‘\(a’\)’ be absorbed by a stronger ‘\(a\)’ and the RA rule is symmetric to LA rule.

The Identity group maintains the same rules of classical logic but also adds a new rule: the \#-Cut rule. This rule express the possibility of cutting formulas with distinct forces since what follows from ‘\(a’\)’ in \#-Cut rule must still follows from a stronger ‘\(a\)’.
Operational rules for the classical logical constants are the same for $?$-formulas and $?$-free formulas, but for the first ones that rules must preserve the order between them. For $?$-formulas we introduce two new Operational rules: $R?$ and $L?$. Intuitively, $R?$ says that you can prove an evidence for $'a'$ if you have $'a'$: $L?$ rule says that you can deny $'a?'$ if you can deny $'a?'$. To make the definition of $'?'$ more precise, it is also necessary to show the relationship of this new logical constant with the old ones. Thus, the distribution rules and $L?$ will complete the definition of the new logical constant $'?'$.

3 The Role of the Labels

In this section we will present a proof example of our sequent calculus to illustrate the role of labels. The labels for the operational rules of the classical part of our sequent calculus were strongly influenced by the labelled natural deduction system presented in [de Queiroz 92]. This labelled natural deduction system is a combination of a functional calculus on the labels and a logical calculus on the formulas. The functional — dynamic — part of the calculus manipulates meta-level informations about the declarative — static — part enabling the control of the proof by appropriate constraints. To our labelled calculus one of the constraints will be the mandatory ordering between $?$-formulas. This must not be violated during the proof.

As we know, a natural deduction system has only two group of rules: introduction and elimination rules for each logical constant. Labels for these rules are composed by constructor and destructor operators to record respectively the introduction and elimination of these logical constants. Since we do not have introduction and elimination but left and right rules in a sequent calculus where the most external logical constant of the detached formula in the lower sequent is eliminated from bottom to top, we must have distinct left and right destructor operators in labels for each logical constant to record the elimination process. The use of those destructor operators will be explained in the example below.

The structure of labels for $?$-formulas is $'x < y'$ where $'x'$ carries the history of the whole deduction of the associated formula — which we call the 'formula identifier' — and $'y'$ is the identifier of the formula immediately stronger. If there is no formula immediately stronger, or if the formula immediately stronger is a $?$-free formula, then the label will be $'(x < T)'$ where $'T'$ stands for the undoubted formula ‘True’. For $?$-free formulas the labels are simply of the form $'x'$ without order. You can think of $'x'$ as a short form of $'(x = T)'$. The labels are extremely important to make clear the hidden provisos related to this order in the rules of our calculus. Unfortunately, we do not have enough space in this paper to present examples using the whole labels. We will deal just with the formula identifier of the related formula without its order and, because of this reason, the rules of the previous section have not been presented with the full labels.

Example: The unlabelled proof is:

\[
\begin{align*}
R \vdash R, M? \vdash M? \\
R, M? \vdash R \land M? & \vdash F? \vdash F? \\
R \vdash R, M? & \vdash \neg F? \vdash F? \\
R \vdash R, M? & \vdash F? \vdash F? \\
R \vdash R, M? & \vdash F? \vdash R \vdash F? \\
R \vdash R, M? & \vdash F? \vdash R \vdash F? \\
\end{align*}
\]

For inserting labels, we will reason backward, i.e., from bottom to top in the proof tree. We first attribute to each atom that appears in a proof a different label variable. In our example we choose $'x'$ for $'R'$, $'y'$ for $'M'$ and $'z'$ for $'F'$. Second, we associate to each root formula a label in normal form. The normal form of labels, i.e. the normal labels, are recursively defined over the structure of the formulas in $L$:

1. The variables $'x, y, z'$ are normal labels for atomics;

799
2. If ‘\(x\)’ is the normal label of ‘\(a\)’, then ‘\(\neg x\)’ is the normal label of ‘\(\neg a\)’;

3. If ‘\(x\)’ is the normal label of ‘\(a\)’, then ‘\(\ldots x\)’ is the normal label of ‘\(\ldots a\)’;

4. If ‘\(x\)’ and ‘\(y\)’ are the normal labels of ‘\(a\)’ and ‘\(b\)’ respectively, then ‘\(\lambda x.y\)’ is the normal label of ‘\(a \rightarrow b\)’;

5. If ‘\(x\)’ and ‘\(y\)’ are the normal labels of ‘\(a\)’ and ‘\(b\)’ respectively, then ‘\(x, y\)’ is the normal label of ‘\(a \land b\)’;

6. If ‘\(x\)’ and ‘\(y\)’ are the normal labels of ‘\(a\)’ and ‘\(b\)’ respectively, then ‘\(x, y\)’ is the normal label of ‘\(a \lor b\)’.

In the following labelled proof we do not repeat a label in an upper sequent if it is not modified by the rule applied to the lower sequent unless it is a label of a leaf formula, i.e., a formula that appears in the identities. The labelled proof tree is:

\[
\frac{\text{var}_2(\lambda x.cz) : R \vdash \text{fst}_2(\text{var}_1(\ast)) : R \quad \ast : M? \vdash \text{snd}_2(\text{var}_1(\ast)) : M?}{\frac{R, M? \vdash \text{var}_1(\ast) : R \land M? \vdash \text{app}_1(\ast, \text{var}_1(\ast)) : F? \vdash F?}{\text{var}_2(\lambda x.cz) : R \vdash \text{var}_1(\lambda x.c(y)) : R \quad \text{var}_2(\lambda x.cz) : R \quad \text{app}_1(\lambda x.c(y), \text{var}_1(\lambda x.c(y))) : M? \quad \text{app}_1(\lambda x.c(y), \text{var}_2(\lambda x.cz)) : M?}}
\]

\[
\frac{R \vdash \text{var}_1(\lambda x.c(y)) : R \quad \text{var}_2(\lambda x.cz) : R \quad R \land M? \vdash F? \vdash F?}{\frac{(\lambda x.c(y)) : R \vdash M?}{(\lambda x.c(y)) : R \land M? \vdash F? \vdash (\lambda x.c(z)) : R \land M? \vdash F?}}
\]

where ‘\(\ast\)’ is ‘\(\lambda x.c(y) > c(z)\)’ and ‘\(\ast\)’ is ‘\(\text{app}_1(\lambda x.c(y), \text{var}_1(\lambda x.c(y)))\)’.

This example illustrates the use of \(R \rightarrow\), \(L\) and \(R \land\) operational rules beyond some structural and identity rules. Each time those rules are applied, the label record the use of the destructor operators ‘\(\text{var}_1(\_\_\_), \text{app}_1(\_\_\_, \_\_\_), \text{app}_2(\_\_\_, \_\_\_), \text{fst}_2(\_\_\_), \text{snd}_2(\_\_\_)\)’ Using labels it is possible to register a sort of justification for each formula that appears in the sequent and reconstruct the root formula from which it depends. It can be quite useful to aid the propagation of a revision in the formulas of the theory when a contradiction arises. Exemplifying, suppose the labelled formula ‘\(\text{var}_2(\text{var}_1(\lambda x.c(y) > c(z))) : R\)’. To rebuild its ancestral root formula, we go forward from top to bottom using the same rules of backward direction as the following:

\[
\ldots \vdash \text{fst}_2(\text{var}_1(\lambda x.c(y) > c(z))) : R
\]

\[
\ldots \vdash \text{var}_1(\lambda x.c(y) > c(z)) : R \land \forall?\lfloor \!
\ldots, \lambda x.c(y) > c(z) : R \land \forall? \vdash \exists? \vdash \ldots
\]

where ‘\(\forall?\)’ and ‘\(\exists?\)’ are formula variables.

Since the label variable ‘\(y\)’ is associated to atom ‘\(M\)’ and ‘\(z\)’ to ‘\(F\)’, we can replace the formula variable ‘\(\forall?\)’ by its instantiated value ‘\(M?\)’, ‘\(\exists?\)’ by its instantiated value ‘\(F?\)’, and we get ‘\(R \land M? \rightarrow F?\)’ as the root formula on which ‘\(R\)’ depends.

In the previous section we have introduced an additional proviso to the identity rules namely: ‘\(x\)’ and ‘\(y\)’ are reducible to the same normal form. We now illustrate how to check this proviso and assume the correctness of the proof through the label reduction sequences of the identity applying the appropriate reduction rules:

\[
\text{app}_1(\lambda x.c(y), \text{var}_1(\lambda x.c(y))) : M? \vdash \text{snd}_2(\text{var}_1(\lambda x.c(y) > c(z))) : M?
\]

The normal form of a conjunction label is a pair ‘\(<y, z>\)’. Its destructor operators are ‘\(\text{fst}_i(\_\_\_, \_\_\_)\)’ and ‘\(\text{snd}_i(\_\_\_, \_\_\_)\)’ (\(i=1, 2\)) to indicate the first and the second projections of that pair respectively. The reduction
rules for \( \text{fst}_i(\_ \_ ) \) and \( \text{snd}_i(\_ \_ ) \), \( i=1,2 \) are: \( \text{fst}_1(y, z) = y \) and \( \text{snd}_1(y, z) = z \), where \( y \) and \( z \) are any arbitrary labels. For an implication label the normal form is a lambda-abstraction \( \lambda y.z \). Its destructor operators are \( \text{var}_i(\_ \_ ) \) and \( \text{app}_i(\_ \_ \_ ) \), \( i=1,2 \), to indicate the lambda-abstraction \( \text{variable} \) and the result of the \text{application} of the abstraction over that variable respectively. The reduction rules for \( \text{var}_1(\_ \_ ) \) and \( \text{app}_1(\_ \_ \_ ) \), \( i=1,2 \) are: \( \text{var}_1(\lambda y.z) = y \) and \( \text{app}_1(\lambda y.z, \text{var}_1(\lambda y.z)) = \text{app}_1(\lambda y.z, y) = z \), where \( y \) and \( z \) are any arbitrary labels. Thus, the reduction sequences are:

1. \( \text{app}_1(\lambda x.\varsigma(y), \text{var}_1(\lambda x.\varsigma(y))) = \text{app}_1(\lambda x.\varsigma(y), x) = \varsigma(y) \) and
2. \( \text{snd}_2(\text{var}_1(\lambda x.\varsigma(y), x, \varsigma(z))) = \text{snd}_2(< x, \varsigma(y), \varsigma(z)> ) = \varsigma(y) \)

ratifying the successful use of the identity rule since the labels have the same normal form \( \varsigma(y) \).

4 Conclusions

The labelled sequent rules for the Logic of Evidence just presented were designed to handle nonmonotonic and paraconsistent reasonings. The labels were introduced to treat side-by-side meta and object level features. The meta level features are revealed by functional labels allowing to reason over the declarative part, i.e., the proof itself. The labels are justifications to the associated formula carrying the history of its proof. An additional role of the labels is the discover of a non cut-free proof just looking to them. This aspect will be emphasized in a coming paper.

Future extensions to this work could be pointed. An obvious one is an extension of our propositional logic to a predicate logic. It will allow a more accurate study over individuals referred by universal and existential quantifiers. Other possible extension could be the treatment of resources through the perspective of relevant or linear resource logics. These issues have to inspire new labels and even new roles for them.

References


