Efficient Theorem Proving for Conditional Logics with Conditional Excluded Middle

Nikola Panic[1], Gian Luca Pozzato[1]

[1]Dipartimento di Informatica, Università di Torino, Italy

Abstract

In this work we introduce a labelled sequent calculus for Conditional Logics admitting the axiom of Conditional Excluded Middle (CEM), rejected by Lewis but endorsed by Stalnaker. We also consider some of its standard extensions. Conditional Logics with CEM recently have received a renewed attention and have found several applications in knowledge representation and artificial intelligence. The proposed calculus improves the only existing one, SeqS, where the condition CEM on conditional models is tackled by means of a simple but computationally expensive process of label substitution. Here we propose an alternative calculus avoiding label substitution, where a single rule deals simultaneously with conditional formulas and the CEM axiom. We have implemented the calculi in Prolog following the "lean" methodology, then we have tested the performances of the prover and compared them with those of CondLean, an implementation of SeqS. The performances are promising and better than those of CondLean, witnessing that the proposed calculus provides an effective improvement with respect to the state of the art.

Keywords

conditional logics, sequent calculi, proof methods, theorem proving

1. Introduction

Conditional logics are extensions of classical logic by means of a binary operator $>$, in order to express conditional implications of the form $A > B$. They have a long history, starting with the seminal works by [1], [2], [3], [4], and [5]. Conditional logics have found an interest in several fields of artificial intelligence and knowledge representation, from reasoning about prototypical properties and non-monotonic reasoning [6, 7, 8, 9], where $A > B$ can be used to formalize that "typically, the As are also Bs" or "in normal circumstances, if A then B", to modeling belief change, knowledge update and revision [10, 11, 12], where the relation with conditional logics is expressed by the so-called *Ramsey's Rule*:

$$(A \circ B) \rightarrow C \text{ holds} \quad \textit{if and only if} \quad A \rightarrow (B > C) \text{ holds}$$

where the operator $\circ$ is any *update* operator satisfying postulates of [13], that are considered the "core" properties for any concrete and plausible operator of belief update. Ramsey's rule means that C is entailed by "A updated by B" if and only if the conditional $B > C$ is entailed by A. In this sense it can be said that the conditional $B > C$ expresses an hypothetical update

CILC 2022: 37th Italian Conference on Computational Logic, June 29 – July 1, 2022, Bologna, Italy
nikola.panic@edu.unito.it (N. Panic); gianluca.pozzato@unito.it (G. L. Pozzato)
0000-0002-3952-4624 (G. L. Pozzato)

of the information A. Moreover, conditional logics have been employed in order to represent conditional sentences that cannot be captured by material implication and, in particular, *counterfactuals* [1], e.g., conditionals of the form "if A were the case, then B would be the case", where A is false, as well as to model hypothetical queries in deductive databases and logic programming [14], causal inference and reasoning about action execution in planning [15, 16], access control policies in security [17].

Similarly to modal logics, the semantics of conditional logics can be defined in terms of possible world structures. In this respect, conditional logics can be seen as a generalization of modal logics (or a type of multi-modal logic) where the conditional operator is a sort of modality indexed by a formula of the same language. However, as a difference with modal logics, a universally accepted semantics for conditional logics lacks and it is the main reason for the underdevelopment of proof-methods and theorem provers. The semantics we consider in this work is the *selection function semantics* [2], where truth values are assigned to formulas depending on a world. Intuitively, the selection function f selects, for a world w and a formula A, the set of worlds $f(w, A)$ which are "most-similar to w" or "closer to w" given the information A. In *normal* conditional logics, the function f depends on the set of worlds satisfying A rather than on A itself, so that $f(w, A) = f(w, A')$ whenever A and A' are true in the same worlds (normality condition). A conditional sentence $A > B$ is true in w whenever B is true in every world selected by f for A and w. It is the normality condition which marks essentially the difference between conditional logics on the one hand, and multimodal logic, on the other (where one might well have a family of $\Box$ indexed by formulas). We believe that it is the very condition of normality what makes it difficult to develop proof systems for conditional logics with the selection function semantics.

Since we adopt the selection function semantics, CK is the fundamental system [2]; it has the same role as the system K (from which it derives its name) in modal logic: CK-valid formulas are exactly those ones that are valid in every selection function model. Extensions are then obtained by imposing restrictions on the selection function. In this work, we focus on the systems equipped with the condition of *Conditional Excluded Middle* (CEM), whose characterizing axioms are of the form

$$(A > B) \lor (A > \neg B)$$

corresponding to the semantic condition that, for each world w and for each formula A, the selection function f selects at most one world for w and A, in other words the cardinality of the selection function is at most 1.

While [1] provides an argument against CEM, essentially based on his treatment of "might" counterfactuals so that both conditionals can be false, [3] provides an argument in favor, intuitively stating that the conditionals can be *indeterminate* but their disjunction is true. Consider the example in [18] and the two counterfactual sentences "if Bizet and Verdi were compatriots, would they be Italian?" and "if Bizet and Verdi were compatriots, would they be not Italian?": Lewis rejects, stating that the two conditionals are intuitively false, whereas Stalnaker endorses it, conjecturing that they are both indeterminate but their disjunction is true. More recently, [18] has provided a general positive argument for CEM, defending the Stalnaker's verdict.

In [19] the authors have introduced a labelled sequent calculus for CK and the extensions with condition CEM, but also ID (identity), MP (conditional modus ponens), and CS (conditional strong centering), as well as most of the combinations of them. The proposed calculi, called SeqS, are modular and, in some cases, optimal, however, for the systems with CEM, a label substitution mechanism is needed in order to deal with the above mentioned condition on the selection function. They have also introduced a Prolog theorem prover, called CondLean, implementing those calculi, whose performances are promising in general, however, due to the label substitution mechanism, they degrade for systems with CEM, especially in finding that a formula is *not* valid.

In this paper we provide a first step in the direction of efficient theorem proving for conditional logics dealing with conditional excluded middle, by tackling the problems of SeqS and CondLean with an alternative calculus (and, as a consequence, an alternative implementation) in which the label substitution mechanism is replaced by a suitable rule for dealing with conditional formulas in these systems. We are able to give cut-free calculi, called SeqS', for CK+CEM and all the extensions with ID and CS. The completeness of the calculi is an immediate consequence of the admissibility of cut. We show that one can derive a decision procedure from the cut-free calculi, providing a constructive proof of decidability of the logics considered. As usual, we obtain a terminating proof search mechanism by controlling the backward application of some critical rules. By estimating the size of the finite derivations of a given sequent, we also obtain a polynomial space complexity bound for these logics.

We have implemented the calculi SeqS' in Prolog following the line of CondLean: our theorem prover is inspired to the "lean" methodology, whose basic idea is to write short programs and exploit the power of Prolog's engine as much as possible. The implementation offer significantly better performances with respect to those of CondLean, allowing us to conclude that the calculi SeqS' can be considered a first, plausible solution to the problem of reasoning in conditional logics with CEM.

The plan of the paper is as follows. In Section 2 we introduce Conditional Logics with Conditional Excluded Middle. In Section 3 we present SeqS', the novel labelled sequent calculi, by emphasizing the differences with SeqS. In Section 4 we describe a Prolog implementation of SeqS', then we conclude in Section 6 with some experimental results witnessing that its performance are better than those of CondLean. and with some pointers to future works.

2. Conditional Logics with Conditional Excluded Middle

In this section we briefly present propositional conditional logics with CEM.

A propositional conditional language $\mathcal{L}$ contains: (i) a set of propositional variables ATM; (ii) the constants $\bot$ and $\top$; (iii) a set of connectives $\neg$ (unary), $\wedge$, $\vee$, $\rightarrow$, $>$ (binary). Formulas of $\mathcal{L}$ as follows:

- $\bot$, $\top$, and the propositional variables of ATM are *atomic formulas*;
- if A and B are formulas, $\neg A$, $A \wedge B$, $A \vee B$, $A \rightarrow B$ and $A > B$ are *complex formulas*.

We define the *selection function semantics* as follows: given a non-empty set of possible worlds $\mathcal{W}$, the selection function f selects, for a world w and a formula A, the set of worlds of $\mathcal{W}$

which are *closer* to w given the information A. A conditional formula $A > B$ holds in a world w if the formula B holds in *all the worlds selected by f for w and A.*

Definition 1 (Selection function semantics). A model is a triple $\mathcal{M} = \langle \mathcal{W}, f, [\,] \rangle$ where:

- $\mathcal{W}$ is a non empty set of *worlds*;
- f is the *selection function*
$$f : \mathcal{W} \times 2^{\mathcal{W}} \longrightarrow 2^{\mathcal{W}}$$
satisfying the condition for *conditional excluded middle*:
$$|\ f(w, [A])\ | \leq 1$$
- $[\,]$ is the *evaluation function*, which assigns to an atom $P \in ATM$ the set of worlds where P is true, and is extended to the other formulas as follows:
  - $[\bot] = \emptyset$;
  - $[\top] = \mathcal{W}$;
  - $[\neg A] = \mathcal{W} \setminus [A]$;
  - $[A \wedge B] = [A] \cap [B]$;
  - $[A \vee B] = [A] \cup [B]$;
  - $[A \rightarrow B] = (\mathcal{W} \setminus [A]) \cup [B]$;
  - $[A > B] = \{w \in \mathcal{W} \mid f(w, [A]) \subseteq [B]\}$.

It is worth noticing that we have defined f taking $[A]$ rather than A (i.e. $f(w,[A])$ rather than $f(w,A)$) as an argument; this is equivalent to define f on formulas, i.e. $f(w,A)$ but imposing that if $[A]$=$[A^{'}]$ in the model, then $f(w, A)$=$f(w, A^{'})$. This condition is called *normality.*

The semantics above characterizes the basic conditional system we consider, called CK+CEM. An axiomatization of this system is given by:

- any axiomatization of classical propositional calculus;
- (CEM) $\quad (A > B) \vee (A > \neg B)$
- (Modus Ponens) $\quad \dfrac{A \quad A \rightarrow B}{B}$
- (RCEA) $\quad \dfrac{A \leftrightarrow B}{(A > C) \leftrightarrow (B > C)}$
- (RCK) $\quad \dfrac{(A_1 \wedge \cdots \wedge A_n) \rightarrow B}{(C > A_1 \wedge \cdots \wedge C > A_n) \rightarrow (C > B)}$

As for modal logics, we can consider extensions of CK+CEM by assuming further properties on the selection function. We consider the following ones:

Logic	Axiom	Model condition
ID	$A > A$	$f(w, [A]) \subseteq [A]$
CS	$(A \wedge B) \rightarrow (A > B)$	$w \in [A] \rightarrow f(w, [A]) \subseteq \{w\}$

The above axiomatization is complete with respect to the semantics [2].

3. A Labelled Sequent Calculus for Conditional Logics with CEM

We introduce SeqS', a sequent calculus for the conditional systems with CEM. The calculi make use of labels to represent possible worlds. We consider a language $\mathcal{L}$ and a denumerable alphabet of labels $\mathcal{A}$, whose elements are denoted by *x, y, z,* .... There are two kinds of labelled formulas:

- *world formulas*, denoted by *x: A*, where $x \in \mathcal{A}$ and $A \in \mathcal{L}$, used to represent that A holds in a world *x*;
- *transition formulas*, denoted by $x \xrightarrow{A} y$, where $x, y \in \mathcal{A}$ and $A \in \mathcal{L}$. A transition formula $x \xrightarrow{A} y$ represents that $y \in f(x, [A])$.

A *sequent* is a pair $\langle \Gamma, \Delta \rangle$, usually denoted with $\Gamma \vdash \Delta$, where Γ and Δ are multisets of labelled formulas. The intuitive meaning of $\Gamma \vdash \Delta$ is: every model that satisfies all labelled formulas of Γ in the respective worlds (specified by the labels) satisfies at least one of the labelled formulas of Δ (in those worlds). Formally, given a model $\mathcal{M} = \langle \mathcal{W}, f, [\,] \rangle$ for $\mathcal{L}$, and a label alphabet $\mathcal{A}$, we consider any *mapping* $I : \mathcal{A} \rightarrow \mathcal{W}$. Let F be a labelled formula, we define $\mathcal{M} \models_I F$ as follows:

- $\mathcal{M} \models_I x$: A if and only if $I(x) \in [A]$
- $\mathcal{M} \models_I x \xrightarrow{A} y$ if and only if $I(y) \in f(I(x), [A])$

We say that $\Gamma \vdash \Delta$ is *valid* in $\mathcal{M}$ if for every mapping $I : \mathcal{A} \rightarrow \mathcal{W}$, if $\mathcal{M} \models_I F$ for every $F \in \Gamma$, then $\mathcal{M} \models_I G$ for some $G \in \Delta$. We say that $\Gamma \vdash \Delta$ is valid in a system, either the basic CK+CEM or any extension of it, if it is valid in every $\mathcal{M}$ satisfying the specific conditions for that system.

The calculi SeqS' are shown in Figure 1. We say that a sequent $\Gamma \vdash \Delta$ is *derivable* if it admits a derivation in SeqS', i.e. a proof tree, obtained by applying backwards the rules of the calculi, having $\Gamma \vdash \Delta$ as a root and whose leaves are all instances of (AX). As usual, the idea is as follows: in order to prove that a formula F is valid in a conditional logic, then one has to check whether the sequent $\vdash x : F$ is derivable in SeqS', i.e. if there is a derivation, obtained by applying backwards the rules, having $\vdash x : F$ as a root.

As a difference with the starting point of this work, namely the sequent calculi SeqS introduced in [19], the calculi SeqS' deal with the CEM condition by means of a second rule whose principal formula is a conditional $A > B$ on the right-hand side of a sequent, in addition to the "standard" one already belonging to the original calculus. The novel rule, called $(CEM^{>})$, is as follows:

$$\frac{\Gamma \vdash \Delta, x \xrightarrow{A} y \qquad \Gamma \vdash \Delta, y : B}{\Gamma \vdash \Delta, x : A > B}\,(CEM^{>})$$

This rule replaces the following rule (CEM) of SeqS:

$$\frac{\Gamma, x \xrightarrow{A} y \vdash \Delta, x \xrightarrow{A} z \qquad (\Gamma, x \xrightarrow{A} y \vdash \Delta)[y/u, z/u]}{\Gamma, x \xrightarrow{A} y \vdash \Delta}\,(CEM)$$

where $\Sigma[x/u]$ is used to denote the multiset obtained from Σ by replacing the label x by u wherever it occurs, and where it holds that $y \neq z$ and $u \notin \Gamma, \Delta$. The basic idea underlying the

CK+CEM

$$(AX)\ \Gamma, x : P \vdash \Delta, x : P \quad (P \in ATM) \qquad (AX)\ \Gamma, x : \bot \vdash \Delta \qquad (AX)\ \Gamma \vdash \Delta, x : \top$$

$$(\neg\mathbf{L})\frac{\Gamma \vdash \Delta, x : A}{\Gamma, x : \neg A \vdash \Delta} \qquad (\neg\mathbf{R})\frac{\Gamma, x : A \vdash \Delta}{\Gamma \vdash \Delta, x : \neg A} \qquad (\vee\mathbf{L})\frac{\Gamma, x : A \vdash \Delta \qquad \Gamma, x : B \vdash \Delta}{\Gamma, x : A \vee B \vdash \Delta}$$

$$(\vee\mathbf{R})\frac{\Gamma \vdash \Delta, x : A, x : B}{\Gamma \vdash \Delta, x : A \vee B} \qquad (\wedge\mathbf{L})\frac{\Gamma, x : A, x : B \vdash \Delta}{\Gamma, x : A \wedge B \vdash \Delta} \qquad (\wedge\mathbf{R})\frac{\Gamma \vdash \Delta, x : A \qquad \Gamma \vdash \Delta, x : B}{\Gamma \vdash \Delta, x : A \wedge B}$$

$$(\rightarrow \mathbf{L})\frac{\Gamma \vdash \Delta, x : A \qquad \Gamma, x : B \vdash \Delta}{\Gamma, x : A \rightarrow B \vdash \Delta} \qquad (>\mathbf{L})\frac{\Gamma, x : A > B \vdash \Delta, x \xrightarrow{A} y \qquad \Gamma, x : A > B, y : B \vdash \Delta}{\Gamma, x : A > B \vdash \Delta}$$

$$(\rightarrow \mathbf{R})\frac{\Gamma, x : A \vdash \Delta, x : B}{\Gamma \vdash \Delta, x : A \rightarrow B} \qquad (>\mathbf{R})\frac{\Gamma, x \xrightarrow{A} y \vdash \Delta, y : B, x : A > B}{\Gamma \vdash \Delta, x : A > B}(y \notin \Gamma, \Delta)$$

$$(CEM^{>})\ \frac{\Gamma \vdash \Delta, x \xrightarrow{A} y \qquad \Gamma \vdash \Delta, y : B}{\Gamma \vdash \Delta, x : A > B} \qquad (\mathbf{EQ})\frac{u : A \vdash u : B \qquad u : B \vdash u : A}{\Gamma, x \xrightarrow{A} y \vdash \Delta, x \xrightarrow{B} y}$$

$$(\mathbf{CS})\frac{\Gamma, x \xrightarrow{A} y \vdash \Delta, x : A \qquad \Gamma[x/u, y/u], u \xrightarrow{A} u \vdash \Delta[x/u, y/u]}{\Gamma, x \xrightarrow{A} y \vdash \Delta}\ (x \neq y, u \notin \Gamma, \Delta) \qquad (\mathbf{ID})\frac{\Gamma, x \xrightarrow{A} y, y : A \vdash \Delta}{\Gamma, x \xrightarrow{A} y \vdash \Delta}$$

Extensions

Figure 1: Rules of sequent calculi SeqS'

new formulation is to generate a new label when dealing with a conditional $x : A > B$ on the right-hand side of a sequent only one time, in order to generate a single world belonging to the selection function of the world represented by x for A, satisfying the semantic condition of having at most one such a world. As an example, Figure 2 shows a derivation of an instance of the characterizing axiom (CEM).

It is easy to observe that the rule $(> \mathbf{R})$ is first applied to $A > B$, introducing the new label y, representing the world selected by the selection function. Then, when the other conditional $A > \neg B$ is taken into account, the rule $(> \mathbf{R})$ is no longer applied, however the new rule $(CEM^{>})$ is applied by selecting the world represented by y as the only one belonging to the "most similar" worlds to the one represented by x given the formula A.

The following basic structural properties hold for all the calculi SeqS' (proofs are similar to those in [19] and omitted to save space.

$$
\dfrac{
  \dfrac{
    \dfrac{}{x \xrightarrow{A} y \vdash x \xrightarrow{A} y, x : A > B, y : B}\,(AX)
    \qquad
    \dfrac{
      \dfrac{}{x \xrightarrow{A} y, y : B \vdash x : A > B, y : B}\,(AX)
    }{x \xrightarrow{A} y \vdash x : A > B, y : B, y : \neg B}\,(\neg \mathbf{R})
  }{x \xrightarrow{A} y \vdash x : A > B, x : A > \neg B, y : B}\,(CEM^{>})
}{
  \dfrac{\vdash x : A > B, x : A > \neg B}{\vdash x : (A > B) \vee (A > \neg B)}\,(\vee \mathbf{R})
}\,(> \mathbf{R})
$$

Figure 2: A derivation of CEM in SeqS'.

Theorem 1 (Height-preserving admissibility of weakening). *If $\Gamma \vdash \Delta$ is derivable in SeqS' with a derivation whose height is h, then also are $\Gamma \vdash \Delta, F$ and $\Gamma, F \vdash \Delta$, with proofs of height $h_1 \leq h$ and $h_2 \leq h$, respectively, where F is any labelled formula.*

Theorem 2 (Height-preserving invertibility of the rules). *If $\Gamma \vdash \Delta$ is derivable in SeqS' with a derivation whose height is h, and $\Gamma \vdash \Delta$ is an instance of the conclusion of a rule R of SeqS', then also $\Gamma' \vdash \Delta'$, where $\Gamma' \vdash \Delta'$ is an instance of one of the premises of R, is derivable in SeqS' with a proof of height $h' \leq h$.*

Theorem 3 (Height-preserving admissibility of contraction). *If $\Gamma \vdash \Delta, F, F$, where F is any labelled formula, is derivable in SeqS' with a derivation whose height is h, then also $\Gamma \vdash \Delta, F$ is derivable in SeqS' with a proof of height $h' \leq h$. If $\Gamma, F, F \vdash \Delta$, where F is any labelled formula, is derivable in SeqS' with a derivation whose height is h, then also $\Gamma, F \vdash \Delta$ is derivable in SeqS' with a proof of height $h' \leq h$.*

The calculi SeqS' are sound and complete for all the systems considered, namely the basic system CK+CEM, as well as the three extensions with ID, CS, and both CS and ID:

Theorem 4 (Soundness and completeness). *Given a conditional formula F, it is valid in a conditional logic with conditional excluded middle if and only if it is derivable in the corresponding calculus of SeqS', that it to say $\models F$ if and only if $\vdash x : F$ is derivable in SeqS'.*

Proof. For the soundness, we have to prove that, if a sequent $\Gamma \vdash \Delta$ is derivable, then the sequent is valid. This can be done by induction on the height of the derivation of $\Gamma \vdash \Delta$. The basic cases are those corresponding to derivations of height 0, that is to say instances of (AX). It is easy to see that, in all these cases, $\Gamma \vdash \Delta$ is a valid sequent. As an example, consider $\Gamma, x : P \vdash \Delta, x : P$: consider every model $\mathcal{M}$ and every mapping I satisfying all formulas in the left-hand side of the sequent, then also $x : P$. This means that $I(x) \in [P]$, but then we have that $\mathcal{M}$ satisfies via I at least a formula in the right-hand side of the sequent, the same $x : P$. For the inductive step, we proceed by considering each rule of the calculi SeqS' in order to check that, if the premise(s) is (are) valid sequent(s), to which we can apply the inductive hypothesis, so is the conclusion. Due to space limitations, we only present the case of the new rule $(CEM^{>})$, for the other rules the proof is similar to the one of SeqS in [19]. Let the considered proof ended as:

$$\frac{(1)\ \Gamma \vdash \Delta, x \xrightarrow{A} y \qquad (2)\ \Gamma \vdash \Delta, y : B}{(3)\ \Gamma \vdash \Delta, x : A > B}\ (CEM^{>})$$

By inductive hypothesis, both (1) and (2) are valid. By absurd, suppose (3) is not, that is to say there exists a model $\mathcal{M}$ and a mapping I satisfying all formulas in Γ but falsifying all formulas in Δ as well as $x : A > B$. Since (1) is valid, since $\mathcal{M}$ and I falsifies all formulas in Δ, necessarily we have that $\mathcal{M} \models_I x \xrightarrow{A} y$, that is to say $I(y) \in f(I(x), [A])$. By the CEM semantic condition, it follows that $(*)\ f(I(x), [A]) = \{I(y)\}$. Analogously, by the validity of (2) we have that $\mathcal{M} \models_I y : B$. If $\mathcal{M} \not\models_I x : A > B$ in (3), there exists a world w such that $w \in f(I(x), [A])$ and $w \notin [B]$, however, since $(*)$, we have that $I(y) = w$, against the validity of (2), and we are done.

The completeness is an easy consequence of the admissibility of the *cut* rule:

$$\frac{\Gamma \vdash \Delta, F \quad F, \Gamma \vdash \Delta}{\Gamma \vdash \Delta}\ (cut)$$

where F is any labelled formula. As usual, the proof proceeds by a double induction over the complexity of the cut formula and the sum of the heights of the derivations of the two premises of cut, in the sense that we replace one cut by one or several cuts on formulas of smaller complexity, or on sequents derived by shorter derivations. We only show one of the paradigmatic cases involving the novel rule $(CEM^{>})$, namely the case in which the cut formula is the principal formulas in both the premises of (cut), and the rules applied to it are $(CEM^{>})$ and $(> \mathbf{L})$. The situation is as follows:

$$\dfrac{\dfrac{(1)\Gamma \vdash \Delta, x \xrightarrow{A} y \quad (2)\Gamma \vdash \Delta, y : B}{(5)\Gamma \vdash \Delta, x : A > B}\ (CEM^{>}) \qquad \dfrac{(3)\Gamma, x : A > B \vdash \Delta, x \xrightarrow{A} y \quad (4)\Gamma, x : A > B, y : B \vdash \Delta}{(6)\Gamma, x : A > B \vdash \Delta}\ (> \mathbf{L})}{\Gamma \vdash \Delta}\ (cut)$$

Since weakening is height-preserving admissible, we can obtain a proof (with a derivation of at most the same height of (5)) for $(5')\ \Gamma \vdash \Delta, x : A > B, y : B$. By inductive hypothesis on the height of the derivations, we can cut (4) and $(5')$, obtaining a derivation of $(7)\ \Gamma, y : B \vdash \Delta$. We can then apply the inductive hypothesis on the complexity of the cut formula to cut (2) and (7), and we are done with a derivation of $\Gamma \vdash \Delta$. The remaining cases are similar to those in [19] and left to the reader.

With the rule (cut) at hand, we show that if a formula F is valid in a conditional logic with CEM, then $\vdash x : F$ is derivable in SeqS'. We proceed by induction on the complexity of the formulas, therefore we show that the axioms are derivable and that the set of derivable formulas is closed under (Modus Ponens), (RCEA), and (RCK). A derivation of axioms **(ID)** and **(CS)** can be obtained as in SeqS [19]. A derivation of (CEM) is provided in Figure 2. For (Modus Ponens), suppose that $\vdash x : A \rightarrow B$ and $\vdash x : A$ are derivable. We easily have that $x : A \rightarrow B, x : A \vdash x : B$ is derivable too. Since cut is admissible, by two cuts we obtain

$\vdash x : B$:

$$\dfrac{\dfrac{x : A \rightarrow B, x : A \vdash x : B \qquad \vdash x : A \rightarrow B}{x : A \vdash x : B}\,(cut) \qquad \vdash x : A}{\vdash x : B}\,(cut)$$

For (RCEA), we have to show that if $A \leftrightarrow B$ is derivable, then also $(A > C) \leftrightarrow (B > C)$ is so. The formula $A \leftrightarrow B$ is an abbreviation for $(A \rightarrow B) \wedge (B \rightarrow A)$. Suppose that $\vdash x : (A \rightarrow B) \wedge (B \rightarrow A)$ is derivable, then also $x : A \vdash x : B$ and $x : B \vdash x : A$ are derivable since rules are height-preserving invertible. We can derive $x : A > C \vdash x : B > C$ as follows:

$$\dfrac{\dfrac{\dfrac{x : A \vdash x : B \qquad x : B \vdash x : A}{x : A > C, x \xrightarrow{B} y \vdash x \xrightarrow{A} y, y : C}\,(\mathbf{EQ}) \qquad x : A > C, x \xrightarrow{B} y, y : C \vdash y : C}{x \xrightarrow{B} y, x : A > C \vdash y : C}\,(> \mathbf{L})}{x : A > C \vdash x : B > C}\,(> \mathbf{R})$$

The other half is symmetric. For (RCK), suppose that (1) $\vdash x : B_1 \wedge B_2 \cdots \wedge B_n \rightarrow C$ is derivable, by the height-preserving invertibility of the rules also $y : B_1, \ldots, y : B_n \vdash y : C$ is derivable. We obtain the following derivation:

$$\dfrac{x \xrightarrow{A} y \vdash x \xrightarrow{A} y \qquad x : A > B_1, y : B_1, \ldots, y : B_n \vdash y : C}{x \xrightarrow{A} y, x : A > B_1, y : B_1, \ldots, y : B_{n-1} \vdash y : C}\,(\Rightarrow \mathbf{L})$$

$$\vdots$$

$$\dfrac{\dfrac{x \xrightarrow{A} y \vdash x \xrightarrow{A} y \qquad x \xrightarrow{A} y, x : A > B_1, \ldots, x : A > B_n, y : B_1 \vdash y : C}{x \xrightarrow{A} y, x : A > B_1, \ldots, x : A > B_n \vdash y : C}\,(> \mathbf{L})}{x : A > B_1, \ldots, x : A > B_n \vdash x : A > C}\,(> \mathbf{R})$$

■

The presence of labels and of the rules ($> \mathbf{L}$), ($\mathbf{ID}$), and ($\mathbf{CS}$), which increase the complexity of the sequent in a backward proof search, is a potential cause of a non-terminating proof search. However, with a similar argument to the one proposed in [19], we can define a procedure that can apply such rules in a controlled way and introducing a finite number of labels, ensuring termination. Intuitively, it can be shown that it is useless to apply ($> \mathbf{L}$) on $x : A > B$ by introducing (looking backward) the same transition formula $x \xrightarrow{A} y$ more than once in each branch of a proof tree. Similarly, it is useless to apply ($\mathbf{ID}$) or ($\mathbf{CS}$)on the same transition $x \xrightarrow{A} y$ more than once in a backward proof search in each branch of a derivation. This leads to the decidability of the given logics:

Theorem 5 (Decidability). *Conditional logics CK+CEM, CK+CEM+ID, CK+CEM+CS, and CK+CEM+ID+CS are decidable.*

We can show that provability in all the conditional logics with CEM considered is decidable in $O(n^2 \ \log n)$ space, the proof is essentially the same as in [19] and can be omitted in order to save space.

4. A Theorem Prover for Conditional Logics with CEM

We have implemented the calculi SeqS' introduced in the previous section (https://gitlab2.educ.di.unito.it/pozzato/condlean4) in order to show that such a calculus can be the base for efficient theorem proving for conditional logics with conditional excluded middle. In order to provide a safe and direct comparison with CondLean [20, 21], as far as we know, the only theorem prover for these logics, we have followed the so-called "lean" methodology, introduced by Beckert and Posegga in the middle of the 90s [22, 23, 24]. Beckert and Posegga have proposed a very elegant and extremely efficient first-order theorem prover, called leanT^AP, consisting of only five Prolog clauses. The basic idea of the "lean" methodology is "to achieve maximal efficiency from minimal means" [22] by writing short programs and exploiting the power of Prolog's engine as much as possible.

We implement each component of a sequent by a list of formulas, partitioned into three sub-lists: atomic formulas, transitions and complex formulas. Atomic and complex formulas are implemented by a Prolog list of the form `[x,a]`, where `x` is a Prolog constant and `a` is a formula. A transition formula $x \xrightarrow{A} y$ is implemented by a Prolog list of the form `[x,a,y]`. Labels are implemented by Prolog constants. The sequent calculi are implemented by the predicate

prove(Cond, Gamma, Delta, Labels, Tree)

which succeeds if and only if $\Gamma \vdash \Delta$ is derivable in SeqS, where `Gamma` and `Delta` are the lists implementing the multisets Γ and Δ, respectively and `Labels` is the list of labels introduced in that branch. `Cond` is a list of pairs of kind $[F, \mathit{Used}]$, where F is a conditional formula `[X,A => B]` and *Used* is a list of transitions $[[X, A_1, Y_1], \ \ldots, [X, A_n, Y_n]]$ such that ($>$ L) has already been applied to $x : A > B$ by using transitions $x \xrightarrow{A_i} y_i$. The list `Cond` is used in order to ensure the termination of the proof search, by applying the restrictions described in the previous section in order to avoid useless applications of the rules. Similar mechanisms are adopted for extensions of the basic system CK+CEM, in order to control the applications of rules (ID) and (CS). `Tree` is an output term: if the proof search succeeds, it matches an implementation of the derivation found by the theorem prover.

Each clause of the `prove` predicate implements one axiom or rule of SeqS'. The theorem prover proceeds as follows. First of all, if $\Gamma \vdash \Delta$ is an axiom, then the goal will succeed immediately by using the clauses for the axioms. If it is not, then the first applicable rule is chosen. The ordering of the clauses is such that the application of the branching rules is postponed as much as possible. Concerning the rules for $>$ on the right-hand side of a sequent, the rule ($>$ R), which introduces a new label in a backward proof search, is first applied to a sequent of the form $\Gamma \vdash \Delta, x : A > B$. If this does not lead to a derivation, the new rule for CEM is then applied.

As an example, the clause for the axiom checking whether the same atomic formula occurs in both the left and the right hand side of a sequent is implemented as follows:

```
prove(_,[LitGamma,_,_],[LitDelta,_,_],_) :-
  member(F,LitGamma),member(F,LitDelta),!.
```

As another example, here is the clause implementing ($>$ L):

```
prove(Cond,[LitGamma,TransGamma,ComplexGamma],
        [LitDelta,TransDelta,ComplexDelta], Labels) :-
  member([X,A => B],ComplexGamma),
  select([[X,A => B],Used],Cond,TempCond),
  member([X,C,Y],TransGamma),
  \+member([X,C,Y],Used),!,
  put([Y,B],LitGamma,ComplexGamma,NewLitGamma,
      NewComplexGamma),
  prove([[[X, A => B],[[X,C,Y] | Used]] | TempCond],
    [LitGamma,TransGamma,ComplexGamma],
    [LitDelta,[[X,A,Y]|TransDelta],ComplexDelta],Labels),
  prove([[[X, A => B],[[X,C,Y] | Used]] | TempCond],
    [NewLitGamma,TransGamma,NewComplexGamma],
    [LitDelta,TransDelta,ComplexDelta],Labels).
```

The predicate `put` is used to put `[Y,B]` in the proper sub-list of the antecedent. The two recursive calls to `prove` implement the proof search on the two premises of the rule.

As a further example, here is the code of the novel rule $(CEM^{>})$:

```
prove(Cond,[LitGamma,TransGamma,ComplexGamma],
        [LitDelta,TransDelta,ComplexDelta], Labels) :-
  select([X,A => B],ComplexDelta,ResComplexDelta),!,
  member([X,_,Y],TransGamma),
  put([Y,B],LitDelta,ResComplexDelta,NewLitDelta,NewComplexDelta),
  prove(Cond,[LitGamma,TransGamma,ComplexGamma],
      [LitDelta,[[X,A,Y] | TransDelta],ComplexDelta], Labels),
  prove(Cond,[LitGamma,TransGamma,ComplexGamma],
      [NewLitDelta,TransDelta,NewComplexDelta], Labels).
```

In order to search a derivation of a sequent $\Gamma \vdash \Delta$, the theorem prover proceeds as follows. First, if $\Gamma \vdash \Delta$ is an axiom, the goal will succeed immediately by using the clauses for the axioms. If it is not, then the first applicable rule is chosen, e.g. if `ComplexDelta` contains a formula `[X,A -> B]`, then the clause for ($\rightarrow$ R) rule is used, invoking `prove` on the unique premise of ($\rightarrow$ R). The prover proceeds in a similar way for the other rules. The ordering of the clauses is such that the application of the branching rules is postponed as much as possible.

In order to check whether a formula is valid in one of the considered system, one has just to invoke the following auxiliary predicate:

pr(Formula)

which wraps the `prove` predicate by a suitable initialization of its parameters.

The theorem prover is available for free download at https://gitlab2.educ.di.unito.it/pozzato/condlean4, where one can also find an updated version of CondLean in order to compute the statistics described in the next section.

5. Statistics

We have tested both CondLean and our theorem prover over

1. a set of randomly generated formulas, either valid or not
2. a set of formulas holding only in systems with CEM

obtaining the following results:

1. over randomly generated formulas, we have observed an improvement of the performances of CondLean of $48,27\%$.
2. over a set of valid formulas we are able to improve the performances of CondLean of $20,57\%$. As an example, running both the provers over the formula

$$(A > (B_1 \vee \dots B_5)) > ((A > B_1) \vee \dots \vee (A > B_5))$$

   our theorem prover is able to build a derivation in 94 ms, against the 266 ms needed by CondLean.

We are currently testing the performances of our implementation over the extensions with ID and CS and we are developing a graphical interface for the prover and we are also providing Prolog files that will allow the user to reproduce a detailed comparison between the two systems in a completely automated way.

The performance of the proposed theorem prover are promising, especially concerning all cases in which it has to answer *no* for a not valid formula: this is justified by the fact that CondLean has to make a great effort in order to explore the whole space of alternative choices in label substitution, operation needed in order to conclude that no derivation exist.

6. Conclusions and Future Works

In this work we have introduced labelled sequent calculi for conditional logics with the axiom of conditional excluded middle (CEM), as well as all the extensions with axioms ID and CS. Our calculi revise those introduced in [19], where a modular labelled sequent calculus SeqS has been introduced for several conditional logics, including those with CEM. We have provided alternative calculi, where the original rule for CEM, based on an expensive mechanism of label substitution, has been replaced by a novel and “standard” rule, called $(CEM^{>})$ specifically tailored for handling conditional formulas $A > B$ in these systems.

We have also implemented the prosed calculi in order to obtain an empirical witness of the fact that our solution improves the one in [19]. We have compared the performances of our theorem prover with those of CondLean, a Prolog implementation of the calculi SeqS. Our implementation is inspired to the “lean” methodology and, in order to focus on CEM, it adopts all the choices of CondLean, essentially just by replacing the rule for conditional excluded middle with a clause implementing the novel $(CEM^{>})$.

In future work we plan to extend the calculi and the implementation to other conditional logics with conditional excluded middle. In particular, our main objective is to include extensions with the axiom MP of conditional modus ponens:

$$(A > B) \rightarrow (A \rightarrow B),$$

whose selection functions must respect the following condition:

$$\text{if } w \in [A], \text{ then } w \in f(w, [A]).$$

This system, as well as its extensions with ID and CS, is not handled by CondLean, since [19] does not show that (cut) is admissible also for them in the calculi SeqS.

Moreover, we aim at implementing a “concrete” theorem prover, starting from the one proposed in this work, implementing state of the art heuristics, data structures and suitable refinements. As already mentioned, we are currently working on extending the set of formulas used in order to obtain further statistics, with the objective of comparing the performances of the proposed theorem prover with those of CondLean.

Acknowledgments

This work has been partially supported by the INdAM - GNCS Project cod. CUP_E55F22000270001 “LESLIE: LogichE non-claSsiche per tooL Intelligenti ed Explainable”.

References

[1] D. Lewis, Counterfactuals, Basil Blackwell Ltd (1973).

[2] D. Nute, Topics in Conditional Logic, Reidel, Dordrecht, 1980.

[3] R. Stalnaker, A theory of conditionals, in: N. Rescher (Ed.), Studies in Logical Theory, Blackwell, 1968, pp. 98–112.

[4] B. F. Chellas, Basic conditional logics, Journal of Philosophical Logic 4 (1975) 133–153.

[5] J. P. Burgess, Quick completeness proofs for some logics of conditionals, Notre Dame Journal of Formal Logic 22 (1981) 76–84.

[6] S. Kraus, D. Lehmann, M. Magidor, Nonmonotonic reasoning, preferential models and cumulative logics, Artificial Intelligence 44 (1990) 167–207.

[7] J. P. Delgrande, A first-order conditional logic for prototypical properties, Artificial Intelligence 33 (1987) 105–130.

[8] N. Friedman, J. Y. Halpern, Plausibility measures and default reasoning, Journal of the ACM 48 (2001) 648–685.

[9] L. Giordano, V. Gliozzi, N. Olivetti, G. L. Pozzato, Analytic tableaux for KLM preferential and cumulative logics, in: G. Sutcliffe, A. Voronkov (Eds.), Logic for Programming, Artificial Intelligence, and Reasoning, 12th International Conference, LPAR 2005, Montego Bay, Jamaica, December 2-6, 2005, Proceedings, volume 3835 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 666–681. URL: https://doi.org/10.1007/11591191_46. doi:10.1007/11591191_46.

[10] G. Grahne, Updates and counterfactuals, Journal of Logic and Computation 8 (1998) 87–117.

[11] L. Giordano, V. Gliozzi, N. Olivetti, Weak agm postulates and strong ramsey test: a logical formalization, Artificial Intelligence 168 (2005) 1–37.

[12] L. Giordano, V. Gliozzi, N. Olivetti, Iterated belief revision and conditional logic, Studia Logica 70 (2002) 23–47.

[13] H. Katsuno, A. O. Mendelzon, On the difference between updating a knowledge base and revising it, in: J. F. Allen, R. Fikes, E. Sandewall (Eds.), Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). Cambridge, MA, USA, April 22-25, 1991, Morgan Kaufmann, 1991, pp. 387–394.

[14] D. M. Gabbay, L. Giordano, A. Martelli, N. Olivetti, M. L. Sapino, Conditional reasoning in logic programming, Journal of Logic Programming 44 (2000) 37–74.

[15] C. B. Schwind, Causality in action theories, Electronic Transactions on Artificial Intelligence (ETAI) 3 (1999) 27–50.

[16] L. Giordano, C. Schwind, Conditional logic of actions and causation, Artificial Intelligence 157 (2004) 239–279.

[17] V. Genovese, L. Giordano, V. Gliozzi, G. L. Pozzato, Logics in access control: a conditional approach, J. Log. Comput. 24 (2014) 705–762. URL: https://doi.org/10.1093/logcom/exs040. doi:10.1093/logcom/exs040.

[18] J. R. G. Williams, Defending conditional excluded middle, Noûs 44 (2010) 650–668. doi:10.1111/j.1468-0068.2010.00766.x.

[19] N. Olivetti, G. L. Pozzato, C. B. Schwind, A Sequent Calculus and a Theorem Prover for Standard Conditional Logics, ACM Transactions on Computational Logics (TOCL) 8 (2007).

[20] N. Olivetti, G. L. Pozzato, Condlean 3.0: Improving condlean for stronger conditional logics, in: B. Beckert (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX 2005, Koblenz, Germany, September 14-17, 2005, Proceedings, volume 3702 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 328–332. URL: https://doi.org/10.1007/11554554_27. doi:10.1007/11554554_27.

[21] N. Olivetti, G. L. Pozzato, Condlean: A theorem prover for conditional logics, in: M. C. Mayer, F. Pirri (Eds.), Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX 2003, Rome, Italy, September 9-12, 2003. Proceedings, volume 2796 of *Lecture Notes in Computer Science*, Springer, 2003, pp. 264–270. URL: https://doi.org/10.1007/978-3-540-45206-5_23. doi:10.1007/978-3-540-45206-5_23.

[22] B. Beckert, J. Posegga, leantap: Lean tableau-based deduction, Journal of Automated Reasoning 15 (1995) 339–358.

[23] B. Beckert, J. Posegga, Logic programming as a basis for lean automated deduction., Journal of Logic Programming 28 (1996) 231–236.

[24] M. Fitting, leantap revisited, Journal of Logic and Computation 8 (1998) 33–47.