Infinite SLaKE-Tableaux

MARCELO FINGER, *Department of Computer Science, University of São Paulo, Rua do Matão 1010, 05508-900, São Paulo, Brazil.*
E-mail: mfinger@ime.usp.br

Abstract

In this work, we explore an algorithm for the Correspondence Theory of Substructural Categorial Logic, known as a SLaKE-tableau. Such algorithm allows us to compute a first-order formula equivalent to a sequent in substructural categorial logic when such sequent generates a finite SLaKE-tableau.

Here we study the cases when such SLaKE-tableau is infinite. We show that it is decidable whether a sequent will generate a finite tableau, presenting a graph-based algorithm for its detection. Our main result is a proof that sequents generating infinite SLaKE-tableaux are *not* equivalent to a first-order formula.

Keywords: Correspondence Theory, KE-Tableaux, SLaKE-Tableaux, Substructural Logics.

1 Introduction

Since Kripke has proposed a possible world semantics for modal logics [5], it has been noted that the presence of certain modal axioms impose specific restrictions on the binary accessibility relation of Kripke frames. Any modal axiom can be translated to a second-order formula and at the heart of *modal correspondence theory* lies the identification of which axioms correspond to some first-order restriction over Kripke frames [10].

In her PhD thesis [6], Natasha Kurtonina has shown that a fragment of substructural logics, known as Categorial Logics [7, 1], can be treated as a modal logic with a Kripke-style semantics based on a *ternary accessibility relation*. Kurtonina showed, using several methods, that some sequents are equivalent to first-order restrictions over ternary frames, while others have no such first-order equivalence.

In a recent work [4], we have proposed an algorithmic method for Categorial Correspondence Theory. This method is based on *SLaKE-tableaux*, and we have shown that a categorial sequent that generates a finite SLaKE-tableau is equivalent to a first-order formula that is obtainable by algorithmic methods.

In this work we extend such result, showing that:

- there is a graph-based algorithm that decides whether a categorial sequent generates an infinite SLaKE-tableau; and
- if a categorial sequent generates an infinite SLaKE-tableau, it is not equivalent to a first-order formula.

Our work is inserted in the tradition of algorithmic correspondence theory for Modal Logics. The most traditional result in this area is the *Sahlqvist-van Benthem Algorithm* that finds a first-order formula corresponding to monomodal formula in the so-called *Sahlqvist* fragment of modal logic [8, 10]. That algorithm works only for that restricted modal fragment, for the problem of finding whether a modal formula has a first-order equivalent is undecidable, a result due to Chagrova. We do not suffer from this restriction because our fragment involves ternary relations (and not binary, as in the monomodal case); also, the semantics of substructural logics further restricts the format of the second-order equivalents of substructural logics. As a result the problem of finding a first-order equivalent is decidable for the fragment dealt with in this paper.

The paper develops as follows. Section 2 presents Substructural Categorial Logics and its semantics based on a ternary accessibility relation. Section 3 introduces SLaKE-tableaux. We concentrate on showing the decidability of finiteness for SLaKE-tableaux in Section 4. Finally, Section 5 shows that sequents associated with infinite tableaux have no first-order equivalent.

2 Substructural Categorial Logics

In this work we deal with the fragment of substructural categorial logic containing a countable set of propositional letters $\mathcal{P} = \{p_0, p_1, \dots\}$ and the usual categorial binary connectives: $/$ (slash), $\backslash$ (backslash) and $\bullet$ (product). The $\bullet$ connective is assumed to be left-associative, that is, $A \bullet B \bullet C \equiv (A \bullet B) \bullet C$. The $/$-connective is the right implication connective, such that from A/B and B (in this order), we infer A; similarly, the $\backslash$-connective is the left implication, so from B and $B\backslash A$ (in this order) infer A.

A sequent is an expression of the form $\Gamma \vdash C$, where Γ is the antecedent and C is the single formula that is contained in the consequent. The antecedents of sequents is a comma-separated binary trees, also left associative, such that an antecedent of the form $A_1, A_2, A_3 \equiv ((A_1, A_2), A_3)$. A sequent calculus presentation for such a logic, known as the *non-associative Lambek Calculus*, has been defined in several places in the literature (e.g. in [7, 1]). But here, we concentrate purely on a modal-like semantic presentation based on *ternary frames*.

We define a *ternary frame* as a pair $\mathrm{F} = (W, R)$, where W is a non-empty set of *possible worlds* and R is a ternary *accessibility relation*. A *ternary model* $\mathcal{M} = (W, R, V)$ consists of ternary frame plus a *valuation* $V : \mathcal{P} \to 2^W$, sending propositional letters to sets of worlds. The semantic interpretation of categorial formulas in the $\{/, \backslash, \bullet\}$-fragment over ternary frames is given below

$$\begin{array}{lll}
\mathrm{F}, V, a \models p & \text{iff} & a \in V(p) \\
\mathrm{F}, V, a \models A/B & \text{iff} & \forall b \forall c (Rcab \wedge \mathrm{F}, V, b \models B \Rightarrow \mathrm{F}, V, c \models A) \\
\mathrm{F}, V, a \models B\backslash A & \text{iff} & \forall b \forall c (Rcba \wedge \mathrm{F}, V, b \models B \Rightarrow \mathrm{F}, V, c \models A) \\
\mathrm{F}, V, a \models B \bullet A & \text{iff} & \exists b \exists c (Rabc \wedge \mathrm{F}, V, b \models A \wedge \mathrm{F}, V, c \models B)
\end{array}$$

We also use the abbreviations:

$$\begin{array}{l}
Rab(cd) =_{def} \exists x (Rabx \wedge Rxcd) \\
Ra(bc)d =_{def} \exists y (Rayd \wedge Rybc)
\end{array}$$

Such abbreviation associates to the left; that is, $Ra(bcd)e =_{def} Ra((bc)d)e$, etc. As usual, we write $\mathrm{F}, a \models A$ when $\mathrm{F}, V, a \models A$ for any valuation V; if a is omitted, this means that the condition holds for any possible world.

A sequent $A_1, \ldots A_n \vdash C$ is valid at a world $a \in W$ in ternary frame F (notation: $\mathrm{F}, a \models (A_1, \ldots A_n \vdash C)$) iff $\mathrm{F}, V, a \models A_1 \bullet \ldots \bullet A_n$ implies $\mathrm{F}, V, a \models C$. This is equivalent to saying that if for some $a_1, \ldots, a_n$, $Ra(a_1 \ldots a_{n-1})a_n$ and, for $1 \leq i \leq n$, $\mathrm{F}, V, a_i \models A_i$, this implies $\mathrm{F}, V, a \models C$.

Note that F can be seen as a first-order model for a language over R and $P_1, \ldots, P_n$, where each P_i is a predicate symbol corresponding to the propositional letter p_i. The notation $(\forall P)\varphi$ indicate the universal closure of all second-order variables in φ, and $(\exists P)\varphi$ the existential one. Following the modal logic tradition, the *standard translation* of a categorial formula A into second-order logic formula is $(\forall P)\forall a ST_a(A)$, where:

$$\begin{array}{lcl} ST_a(p_i) & = & P_i(a) \\ ST_a(A \bullet B) & = & \exists bc(Rabc \wedge ST_b(A) \wedge ST_c(B)) \\ ST_a(B/A) & = & \forall bc(Rcab \wedge ST_b(A) \Rightarrow ST_c(B)) \\ ST_a(A\backslash B) & = & \forall bc(Rcba \wedge ST_b(A) \Rightarrow ST_c(B)) \end{array}$$

The second-order quantification is over the relevant predicate symbols and reflects all relevant valuation in a frame. So every categorial sequent $A \vdash C$ corresponds to a second-order formula such that $\mathrm{F} \models (A_1, \ldots, A_n \vdash C)$ iff

$$\mathrm{F} \models (\forall P)\forall a_1 \ldots a_n a\ (ST_{a_1}(A_1) \wedge \ldots \wedge ST_{a_n}(A_n) \wedge R(a_1 \ldots a_n)a \Rightarrow ST_a(C))$$

The goal of correspondence theory is to know when such a second-order formula defines a *first-order* frame property, that is, if there is a first-order formula ψ such that

$$\mathrm{F} \models (A_1, \ldots, A_n \vdash C) \textit{ iff } \mathrm{F} \models \psi$$

It is the computation of such a property, when it exists, that we investigate next by means of SLaKE-tableaux.

3 Substructural Labelled KE Tableaux

Substructural Labelled KE (SLaKE) tableaux are the main proof theoretical equipment we use in the generation of correspondence formulas for substructural sequents.

D'Agostino has shown in [3] that analytic tableaux, in the style proposed by Smullyan [9], cannot polynomially simulate truth tables and in some cases perform much worse than them. To avoid such problems in a principled way, KE-tableaux were introduced.

The use of KE tableaux for substructural logics have been proposed in [2], by attaching a label to the signed formula, as in $T\ A : a$. We use this idea, but without following its labelling discipline. Instead, we simply add a new label at each new node of the tableau. Formally, let $\mathcal{L}$ be a countable set of labels, let A be a categorial formula; then for every $a \in \mathcal{L}$, the formulas $T\ A : a$ and $F\ A : a$ are *signed labelled formulas*.

Each finite SLaKE-tableau is associated with a *correspondence formula*. The original sequent is associated with an initial tableau and with a second-order formula. There are several *linear expansion rules* and a single *branching rule* based on the Principle of Bivalence. Each of the tableau expansion rules is associated with an expansion of the correspondence formula of the form $\sharp_i := \varphi(R, B_1, \ldots, B_n, \sharp_{i+1})$, where R is the ternary accessibility relation, $B_1, \ldots, B_n$ are the formulas generated in the expansion, and $\sharp$ is the "substitution place" for next expansion and can be read simply as *truth*. The tableau rules for SLaKE-tableaux are illustrated in Figure 1.

Name	*Type*	*SLaKE Expansion*	*Formula Expansion*
$(T/)$	(linear-$\forall$)	$\dfrac{\begin{array}{c} T\ A/B : a \\ T\ B : b \end{array}}{T\ A : c \text{ (new } c)}$	$\sharp_i := \forall c(Rcab \Rightarrow (ST_c(A) \wedge \sharp_{i+1}))$
$(F/)$	(linear-$\exists$)	$\dfrac{F\ A/B : a}{\begin{array}{c} T\ B : b \text{ (new } b) \\ F\ A : c \text{ (new } c) \end{array}}$	$\sharp_i := \exists b \exists c(Rcab \wedge ST_b(B) \wedge \neg ST_c(A) \wedge \sharp_{i+1})$
$(T\backslash)$	(linear-$\forall$)	$\dfrac{\begin{array}{c} T\ B\backslash A : a \\ T\ B : b \end{array}}{T\ A : c \text{ (new } c)}$	$\sharp_i := \forall c(Rcba \Rightarrow (ST_c(A) \wedge \sharp_{i+1}))$
$(F\backslash)$	(linear-$\exists$)	$\dfrac{F\ B\backslash A : a}{\begin{array}{c} T\ B : b \text{ (new } b) \\ F\ A : c \text{ (new } c) \end{array}}$	$\sharp_i := \exists b \exists c(Rcba \wedge ST_b(B) \wedge \neg ST_c(A) \wedge \sharp_{i+1})$
$(T\bullet)$	(linear-$\exists$)	$\dfrac{T\ A \bullet B : a}{\begin{array}{c} T\ A : b \text{ (new } b) \\ T\ B : c \text{ (new } c) \end{array}}$	$\sharp_i := \exists b \exists c(Rabc \wedge ST_b(A) \wedge ST_c(B) \wedge \sharp_{i+1})$
$(F\bullet)$	(linear-$\forall$)	$\dfrac{\begin{array}{c} F\ A \bullet B : a \\ T\ A : b \end{array}}{F\ B : c \text{ (new } c)}$	$\sharp_i := \forall c(Rabc \Rightarrow (\neg ST_c(B) \wedge \sharp_{i+1}))$
(PB)	(branch)	$\dfrac{}{T\ A : x \qquad F\ A : x}$	$\sharp_i := \forall x((ST_x(A) \wedge \sharp^1) \vee (\neg ST_x(A) \wedge \sharp^2))$

FIG. 1. SLaKE rules

In each linear rule in Figure 1, the formulas above the horizontal line are the *premises* of the rule, and those below it are the *conclusions* of the rule. There are one-premised and two-premised rules, but each rule has exactly one premise that is a *compound* formula, which is called the *main premise*; other premises are called *auxiliary*. Two-premised rules are *universal* and one-premised rules are *existential*. If either of the conclusions of an existential rule is present on the current branch, it is not added again with a new label. Universal rules always generate a new conclusion.

The last rule in Figure 1 is the Principle of Bivalence (PB) branching rule. It is only applied for a formula A following the *branching heuristics*:

> PB is used for a formula A that serves as an auxiliary premise for a $\forall$-rule; PB is only applied if there is a unused $\forall$-main premise but no other linear expansion is possible.

The main premises that trigger the application of PB for A are: $F\ A \bullet B$, $T\ A\backslash B$ and $T\ B/A$. The corresponding $\forall$-rule will be applicable on the $T\ A$ branch. Only subformulas of the original sequent will be introduced by PB. It introduces two "substitution places" in the correspondence formula, $\sharp^1$ and $\sharp^2$, one for each new branch. Each active branch in a SLaKE tableau always has exactly one substitution place. Substitution places guarantee that each formula introduced in the correspondence formula will be in the scope the correct quantifiers.

A sequent of the form $A_1, \ldots, A_n \vdash C$ is transformed into the initial SLaKE-tableau:

$$\begin{array}{c} T\ A_1 : a_1 \\ \vdots \\ T\ A_n : a_n \\ F\ C : a \end{array}$$

Since the tableau is a refutation method, this induces the correspondence formula:

$$\neg\varphi = \neg\exists a a_1 \ldots a_n [ST_{a_1}(A_1) \wedge \ldots \wedge ST_{a_n}(A_n) \wedge \neg ST_a(C) \wedge Ra(a_1 \ldots a_{n-1})a_n \wedge \sharp]$$

A single premised sequent $A \vdash C$ generates the initial tableau containing $T\ A : a$ and $F\ C : a$, with the initial correspondence formula $\neg\varphi(\sharp_1) = \neg\exists a(ST_a(A) \wedge \neg ST_a(C) \wedge \sharp_1)$. We could extend the method for sequents with empty antecedents, but we do not pursue this topic here.

The aim of the SLaKE-tableau construction is *not* to close every tableau branch, but to expand each tableau branch until no more expansions are possible. Each expansion step will also give us a new version of the correspondence formula. If we can finitely expand all tableau branches, a valuation for the atomic formula is constructed, so that we obtain a first-order formula by substituting in the final formula the evaluated values. It is also possible that there will be some infinite branches (something that would not happen in simple propositional tableaux).

Formally, let the countable set of labels $\mathcal{L}$ be linearly ordered by $<_{\mathcal{L}}$. A *SLaKE-saturated set* Γ is a set of labelled signed formulas such that, with respect to the rules of Figure 1:

(a) If the premise of a $\exists$-rule is in Γ, each of its consequence is in Γ for some label.

(b) If the premises of a $\forall$-rule are in Γ, its consequence is in Γ with a label *bigger than those of its premises*, according to $<_{\mathcal{L}}$.

(c) For each compound formula in Γ that is a main premise of a $\forall$-rule, there must be in Γ either a an auxiliary premise or the opposite of it.

The expansion of a tableau aims at constructing branches that are SLaKE-saturated sets. Items (a) and (b) correspond to normal branch expansion. The restriction on the order of the consequence labels in (b) rises the possibility of having infinite SLaKE-saturated sets. Item (c) is the branching heuristics.

The expansion of a tableau is thus the stepwise construction of a *saturated counter-model* for the input sequent. Each step generates a second-order formula $\neg\varphi_{i+1}(\sharp_{i+1})$ that is equivalent to $\neg\varphi_i(\sharp_i)$. At the end of a finite expansion, a correspondence formula will be built from a suitable valuation, called the *canonical valuation*.

DEFINITION 3.1
Let $X \in \{T, F\}$; define $\bar{X}$ such that $\bar{T} = F$ and $\bar{F} = T$. For each SLaKE atomic formula $X\ p : x$ in the tableau, define:

$$O(X\ p : x) = \{y \mid \bar{X}\ p : y \text{ precedes X p:x}\}$$

For every atomic p, define the *canonical valuation* of $P(x)$ implicitly as:

$$\bigwedge_{Xp:x} ((P(x) \leftrightarrow \bigwedge_{y \in O(Tp:x)} x \neq y) \wedge (\neg P(x) \leftrightarrow \bigwedge_{y \in O(Fp:x)} x \neq y))$$

The canonical valuation is the *minimal condition* a countermodel must satisfy to falsify the input sequent:

- all the atomic conditions determined by the tableau must hold (see Lemma 5.1); and
- no formula can be both true and false at the same world.

To obtain a first-order formula we need to substitute the *canonical* valuation into $\neg\varphi$. If the tableau saturates, such a valuation/substitution satisfies all compound formulas. The first-order formula $\neg\psi$ thus obtained is obviously implied by $\neg\varphi$. We know that

FACT 3.2 ([4])
If a sequent produces a finite SLaKE-tableau, then it is equivalent to a first-order restriction over ternary frames. [1]

We want to show the converse of such statement, namely that infinite tableaux have no first-order equivalent. Let us first see some examples of SLaKE tableaux.

3.1 Finite SLaKE-Tableau

Consider the sequent $A/B \vdash B\backslash A$. Its associated SLaKE tableau is:

1.	$T\ p/q : a$		
2.	$F\ q\backslash p : a$		$\neg\varphi = \neg\exists a(ST_a(p/q) \wedge \neg ST_a(q\backslash p) \wedge \sharp_1)$
3.	$T\ q : b$	from 2	
4.	$F\ p : c$	from 2	$\sharp_1 := \exists b \exists c(Rcba \wedge Q(b) \wedge \neg P(c) \wedge \sharp_2)$
5.	$T\ p : d$	from 1,4	$\sharp_2 := \forall d(Rdab \Rightarrow P(d) \wedge \sharp_3)$

Initially, we expand line 2, and simultaneously, using the semantics of $q\backslash p$, we expand the correspondence formula substituting $\sharp_1$ into it. We then use lines 1 and 3 for another expansion, obtaining $\sharp_2$. At this point, the tableau is saturated and closed (remember our main goal is not to close a tableau, but to saturate every branch of it). Make $\sharp_3 := \top$. We have thus built the second-order formula:

$$\neg\varphi = \neg\exists a(ST_a(p/q) \wedge \neg ST_a(q\backslash p) \wedge \exists b \exists c(Rcba \wedge Q(b) \wedge \neg P(c) \wedge \forall d(Rdab \Rightarrow P(d))))$$

[1] This result has not been published yet, the the results in this paper are not dependent on it but are intended to complement it.

The formula $\neg\varphi$ is equivalent to the original sequent. We construct the following implicit canonical valuation for the atoms in $\neg\varphi$, by forcing that no world both contain a T- and an F-signed atom:

$$Q(b) \leftrightarrow \top \qquad\qquad (\neg P(c) \leftrightarrow \top) \wedge (P(d) \leftrightarrow d \neq c)$$

Saturation guarantees that the compound formulas in $\neg\varphi$ can be substituted by $\top$. By substituting each second-order variable in $\neg\varphi$ by their canonical valuation, we obtain the first-order formula

$$\neg\exists a \exists b \exists c(\top \wedge \top \wedge Rcba \wedge \exists b \exists c(Rcba \wedge \top \wedge \top \wedge \forall d(Rdab \Rightarrow d \neq c)))$$

which is equivalent to $\forall a \forall b \forall c(Rcba \Rightarrow Rcab)$, meaning that R must have the *commutativity* property for its second and third positions. It is easy to verify that any model satisfying such a property also satisfy the original sequent, and therefore it is *equivalent* to it.

Now let us see an example with branching. For that, consider Peirce's Axiom $(p\backslash q)\backslash p \vdash p$.

$$\begin{array}{cc|cc}
\multicolumn{4}{c}{1.\quad T\ (p\backslash q)\backslash p : a} \\
\multicolumn{4}{c}{2.\quad F\ p : a} \\
\hline
3(i). & T\ p\backslash q : b & 3(ii). & F\ p\backslash q : b \\
\hline
4(i). & T\ p : c & 4(ii). & T\ p : e \\
\cline{1-2}
5(i). & T\ q : d & 5(ii). & F\ q : f
\end{array}
\qquad
\begin{array}{ll}
\multicolumn{2}{c}{\neg\varphi = \neg\exists a(ST_a((p\backslash q)\backslash p) \wedge \neg P(a) \wedge \sharp_1)} \\
\hline
\multicolumn{2}{c}{\sharp_1 := \forall b((ST_b(p\backslash q) \wedge \sharp_2^i) \vee (\neg ST_b(p\backslash q) \wedge \sharp_2^{ii}))} \\
\hline
\sharp_2^i = \forall c(Rcba \Rightarrow (P(c) \wedge \sharp_3^i)) & \\
\cline{1-1}
\sharp_3^i = \forall d(Rdcb \Rightarrow (Q(d) \wedge \sharp_4^i)) & \sharp_2^{ii} = \exists e \exists f(Rfeb \wedge P(e) \wedge \neg Q(f) \wedge \sharp_3^{ii})
\end{array}$$

The branches are independently developed. The left branch will cause substitutions in $\sharp_2^i$ and the right branch will cause substitutions in $\sharp_2^{ii}$. If we were looking for a closed tableau, we could stop the development of the left branch at $4(i)$; however, our aim here is to obtain a *SLaKE-saturated* set at each branch, so we proceed to obtain $5(i)$. The correspondence formula obtained after both branches are saturated is:

$$\begin{array}{ll}
\forall a \exists b[& \neg(ST_a((p\backslash q)\backslash p) \wedge \neg P(a) \wedge ST_b(p\backslash q) \wedge \forall c(Rcba \Rightarrow (P(c) \wedge \forall d(Rdcb \Rightarrow Q(d)))))] \\
 & \wedge \\
\forall a \exists b[& \neg(ST_a((p\backslash q)\backslash p) \wedge \neg P(a) \wedge \neg ST_b(p\backslash q) \wedge \exists e \exists f(Rfeb \wedge P(e) \wedge \neg Q(f)))]
\end{array}$$

The canonical valuation is developed such that no atom is both false and true at a world; to guarantee the right scoping of variables, a world is forced to be different from those opposite ones preceding it in the tableau. This results in the valuation:

$$\begin{array}{c}
(\neg P(a) \leftrightarrow \top) \wedge (P(c) \leftrightarrow c \neq a) \wedge (P(e) \leftrightarrow e \neq a) \\
(Q(d) \leftrightarrow \top) \wedge (\neg Q(f) \leftrightarrow \top)
\end{array}$$

The first-order correspondence formula thus obtained is:

$$\forall a \exists b(Raba \wedge \forall ef(Rfeb \Rightarrow e = a))$$

3.2 *Infinite SLaKE-Tableau*

It is not always the case that a tableau branch can be finitely saturated. In those cases we cannot apply the method above, so we do not get a first-order formula. For example, consider the tableau for $p/p \vdash p\backslash p$:

$$
\begin{array}{lll}
1. & T\ p/p : a & \\
2. & F\ p\backslash p : a & \\
\hline
3. & T\ p : b & \text{from 2} \\
4. & F\ p : c & \text{from 2} \\
5. & T\ p : d & \text{from 1 and 3} \\
6. & T\ p : e & \text{from 1 and 5} \\
7. & T\ p : f & \text{from 1 and 6} \\
 & \vdots &
\end{array}
$$

It is clear that the aim of constructing a SLaKE-saturated leads in this case to an infinite tableau (with a single branch). Another example is $(A\backslash A)\backslash A \vdash A$, which contains one finite and one infinite branch:

$$
\begin{array}{ll}
1. & T\ (p\backslash p)\backslash p : a \\
2. & \quad F\ p : a
\end{array}
$$

$$
\begin{array}{ll@{\qquad\qquad}ll}
3i. & T\ p\backslash p : b & 3ii. & F\ p\backslash p : b \\
4i. & T\ p : c & 4ii. & T\ p : e \\
5i. & T\ p : d \text{ from } 3i \text{ and } 4i & 5i. & F\ p : f \\
6i. & T\ p : e \text{ from } 3i \text{ and } 5i & & \\
 & \vdots & &
\end{array}
$$

It was shown in [6] that those sequents have no first-order correspondence.

4 Detecting Infinite Branches

We can detect infinite branches of SLaKE-tableaux in the categorial fragment in the following way. For each branch $\mathcal{B}$ in a tableau, define its *dependency graph* $\mathcal{G}_{\mathcal{B}} = (N, E)$, where a node in N is a set of signed unlabelled formulas used as joint premises in an expansion step in $\mathcal{B}$, that is:

$$N = \{\{X_1 A_1, \ldots, X_n A_n\} |\ \ X_i \in \{T, F\} \text{ and } X_1\, A_1 : a_1, \ldots, X_n\, A_n : a_n \in \mathcal{B} \text{ are used as joint premises of the same derivation step in } \mathcal{B}\text{'s expansion } \}$$

According to the expansion rules in Figure 1, the nodes will be singletons or have two elements. There is a directed edge from $n_1 = \{X_1 A_1, \ldots, X_n A_n\}$ to $n_2 \ni X\, A$ iff in $\mathcal{B}$'s construction there was an expansion rule with premises $X_1\, A_1 : a_1, \ldots, X_n\, A_n : a_n$ and consequence $X\, A : a$.

THEOREM 4.1
A branch $\mathcal{B}$ in a SLaKE tableau over the categorial fragment is infinite iff $\mathcal{G}_{\mathcal{B}}$ has a cycle.

A tableau is *non-deterministic* if at some point in its expansion two or more rules were applicable.

COROLLARY 4.2
If one expansion of a non-deterministic branch is infinite, then any other possible expansion will also be infinite.

COROLLARY 4.3
It is decidable whether the expansion of a finite set of SLaKE-formulas will lead to an infinite branch.

5 The Second-Orderness of Infinite SLaKE-Tableaux

The aim of this section is to show that if a SLaKE-tableau has an infinite branch, then its corresponding second-order formula $\neg\varphi$ cannot be equivalent to a first-order formula. For that we use an auxiliary construction.

Parallel to the construction of a SLaKE-saturated set, we build a R-configuration ρ. The set ρ is initially empty and is built during the tableau's expansion:

(a) If a $\exists$-rule generates an expansion of the form $\exists xy(R(x,y,z) \wedge \ldots)$, add $R(x,y,z)$ to ρ.
(b) If a $\forall$-rule generates an expansion of the form $\forall xy(R(x,y,z) \wedge \ldots \Rightarrow \ldots)$, add $R(x,y,z)$ to ρ.

The configuration ρ allows us to construct a model for the SLaKE-saturated set, which will be a countermodel to the input sequent.

Suppose we have a saturated branch $\mathcal{B}$, that may be finite or infinite, with associated SLaKE-saturated set Γ and R-configuration ρ. A *$\mathcal{B}$-model* is any model $\mathcal{M}_\mathcal{B} = (W, R, V_\mathcal{B})$ such that:

$$\begin{array}{l} W \supseteq \{x \mid x \text{ is a lable ocurring in } \Gamma\} \\ R \supseteq \{\langle x,y,z\rangle \mid Rxyz \in \rho\} \\ \text{If } F\ p : x \in \Gamma \text{ then } x \notin V_\mathcal{B}(p) \\ \text{If } T\ p : x \in \Gamma \text{ then } x \in V_\mathcal{B}(p) \end{array}$$

Let $X\ A : a \in \Gamma$, $X \in \{T, F\}$. We say that $\mathcal{M}_\mathcal{B}$ *satisfies* $X\ A : a$ if $\mathcal{M}_\mathcal{B}, a \models A$ iff $X = T$.

LEMMA 5.1
Let Γ be a SLaKE-saturated set corresponding to a branch $\mathcal{B}$, and let $\mathcal{M}_\mathcal{B}$ be a $\mathcal{B}$-model. For every $X\ A : x \in \Gamma$ then $\mathcal{M}_\mathcal{B}$ satisfies $X\ A : x$.

LEMMA 5.2
Consider a sequent $A_1, \ldots, A_n \vdash C$ whose tableau contains a saturated branch $\mathcal{B}$. Let $\neg\varphi$ be the sequent's corresponding second-order formula. Then, for any $\mathcal{B}$-model, $\mathcal{M}_\mathcal{B}$, $\mathcal{M}_\mathcal{B} \not\models \neg\varphi$.

Now let us consider a sequent $A_1, \ldots, A_n \vdash C$ whose saturated SLaKE-tableau contains an infinite branch $\mathcal{B}$ and whose second-order corresponding formula is $\neg\varphi$.

For each step i in the construction of the infinite branch $\mathcal{B}$, let $\mathcal{B}_i$ be the set of formulas contained in the tableau at step i. We define a theory τ_i inductively as:

$$\begin{array}{ll} W_i = \{x \mid x \text{ is a lable ocurring in } \mathcal{B}_i\} & \text{the world at } i \\ \alpha_i = \bigwedge\{x \neq y \mid x, y \in W_i; x \neq y\} & \text{unique names for worlds} \\ \rho_i = \bigwedge\{Rxyz \in \rho \mid x, y, x \in W_i\} & \rho \text{ restricted to } W_i \\ V_i = \bigwedge\{P(x) \mid T\ p : x \in \mathcal{B}_i\} \wedge \bigwedge\{\neg Q(x) \mid F\ q : x \in \mathcal{B}_i\} & V \text{ restricted to } W_i \\ \tau_i = \neg\varphi \wedge V_i \wedge \alpha_i \wedge \rho_i & \end{array}$$

LEMMA 5.3
At every step i there is a model for τ_i.

THEOREM 5.4
If a categorial sequent generates a SLaKE-tableau with an infinite branch, than it is not equivalent to a first-order formula.

PROOF. Assume that the tableau corresponding formula $\neg\varphi$ is equivalent to a first-order formula. By Corollary 4.2 any expansion of the input sequent will generate an infinite branch. For any such infinite branch $\mathcal{B}$, generate the infinite sequence $\{\tau_i\}_{i\in\mathbb{N}}$, where each τ_i is equivalent to a first-order formula. By Lemma 5.3, any finite subset of $\{\tau_i\}_{i\in\mathbb{N}}$ has a model (just take the model for the largest i). However, any model of $\{\tau_i\}_{i\in\mathbb{N}}$ is a $\mathcal{B}$-model and by Lemma 5.2 it cannot satisfy $\neg\varphi$, so $\{\tau_i\}_{i\in\mathbb{N}}$ has no models.

This contradicts the compactness theorem for first-order logic, so $\neg\varphi$ cannot be equivalent to a first-order formula. ∎

Acknowledgements

The author was partly supported by the Brazilian Research Council (CNPq), grant PQ 300597/95-5.

References

[1] Bob Carpenter. *Type-Logical Semantics*. MIT Press, 1997.

[2] M. D'Agostino and D. Gabbay. A Generalization of Analytic Deduction via Labelled Tableaux, part I: Basic Substructural Logics. *Journal of Automated Reasoning*, 13:243–281, 1994.

[3] Marcello D'Agostino. Are tableaux an improvement on truth-tables? — cut-free proofs and bivalence. *Journal of Logic, Language and Information*, 1:235–252, 1992.

[4] M. Finger. Algorithmic correspondence theory for substructural categorial logic. Submitted for publication, 2000.

[5] Saul Kripke. Semantical analysis of modal logic I. *Zeitschrift fur Mathematische Logik und Grundlagen der Mathematik.*, 9(67–96), 1963.

[6] Natasha Kurtonina. *Frames and Labels — A Modal Analysis of Categorial Inference*. PhD thesis, Research Institute for Language and Speech (OTS), Utrecht, and Institute of Logic, Language and Information (ILLC), Amsterdam, 1994.

[7] M. Moortgat. Categorial type logics. In J. Van Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*, pages 93–178. Elsevier North-Holland/The MIT Press, 1997.

[8] H. Sahlqvist. Completeness and correspondence in the first- and second-order semantics for modal logic. In *Proceedings of the 3rd Scandinavian Logic Symposium*, 1975.

[9] Raymond M. Smullyan. *First-Order Logic*. Springer-Verlag, 1968.

[10] J. van Benthem. Correspondence theory. In *Handbook of Philosophical Logic*, volume II, pages 167–248. D. Reidel Publishing Company, 1984.