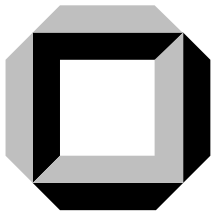
Free Variable Tableaux for Propositional Modal Logics

Bernhard Beckert
Rajeev Goré

Interner Bericht 41/96



Universität Karlsruhe
Fakultät für Informatik

Free Variable Tableaux for Propositional Modal Logics

Bernhard Beckert* Rajeev Goré**

http://i12www.ira.uka.de/modlean

* Address: University of Karlsruhe, Institute for Logic, Complexity and Deduction Systems, D-76128 Karlsruhe, Germany. Email: beckert@ira.uka.de.

** Address: Automated Reasoning Project, Australian National University, Canberra, ACT, 0200, Australia. Email: rpg@arp.anu.edu.au.

Abstract

We present a sound, complete, modular and *lean* labelled tableau calculus for many propositional modal logics where the labels contain "free" and "universal" *variables*. Our "lean" Prolog implementation is not only surprisingly short, but compares favourably with other considerably more complex implementations for modal deduction.

1 Introduction

Free variable semantic tableaux are a well-established technique for first-order theorem proving—both theoretically and practically. Free variable quantifier rules [18, 7] are crucial for efficiency since free variables act as a meta-linguistic device for tracking the eigenvariables used during proof search.

Traditional tableau-based theorem provers developed during the last decade for first-order logic have been complex and highly sophisticated, typified by systems like Setheo [15] and ${}_3T^AP$ [1]. On the other hand, free variable tableaux, and their extensions like universal variable tableaux, have been used successfully for *lean* Prolog implementations, as typified by $\mathsf{lean}T^AP$ [2]. A "lean" implementation is an extremely compact (and efficient) program that exploits Prolog's built-in clause indexing scheme and backtracking mechanisms instead of relying on elaborate heuristics. Such compact lean provers are much easier to understand than their more complex stablemates, and hence easier to adapt to special needs.

Simultaneously, Kanger's meta-linguistic indices for non-classical logics [14] have been generalised by Gabbay into Labelled Deductive Systems [9]. And Massacci [16] and Russo [19] have recently shown the utility of using *ground* labels for obtaining *modular* modal tableaux and natural deduction systems (respectively); see [10] for an introduction to labelled modal tableaux.

By allowing labels to contain free (and universal) variables, we obtain *efficient* and *modular* tableaux systems for all the 15 basic *propositional* modal logics. Furthermore, our leanTAP style implementation compares favourably with existing fast implementations of modal tableau systems like LWB [12].

Our object language uses *labelled formulae* like $\sigma : A$, where σ is a label and A is a formula, with intuitive reading "the possible world σ satisfies the formula A"; see [6, 17, 10] for details. Thus, $1 : \Box p$ says that the possible world 1 satisfies the formula $\Box p$. Our box-rule then reduces the formula $1 : \Box p$ to the labelled formula $1.(x) : p$ which contains the *universal* variable x in its label and has an intuitive reading "the possible world $1.(x)$ satisfies the formula p". Since different instantiations of x give different labels, the labelled formula $1.(x) : p$ effectively says that "all successors of the possible world 1 satisfy p", thereby capturing the usual Kripke semantics for $\Box p$ (almost) exactly. But the possible world 1 may have *no* successors; so we enclose the variable in parentheses and read $\sigma : A$ as "for all instantiations of the variables in σ, if the world corresponding to that instantiation of σ exists then the world satisfies the formula A".

Similar approaches using labels containing variables have been explored by Governatori [11] and D'Agostino et al. [4]. But D'Agostino et al. relate the labels to modal algebras, instead of to first-order logic as we do. And whereas Governatori uses string unification over labels to detect complementary formulae, we use Prolog's matching, since string unification cannot be implemented in a lean way. Our variables are of a simpler kind: they capture all *immediate* children of a possible world (in a rooted tree model), but do not capture *all R-successors*; see [16, 10]. As a consequence, we can make extensive use of Prolog features like unification and backtracking in our implementation. Note, however, that a non-lean extension of our calculi using string unification is perfectly feasible.

The following techniques, in particular, are crucial:

Free variables: Applying the traditional ground box-rule requires guessing the correct eigenvariables. Using (free) variables in labels as "wildcards" that get instantiated "on demand" during branch closure allows more intelligent choices of these eigenvariables. To preserve soundness for worlds with no R-successors, variable positions in labels must be conditional.

Universal variables: Under certain conditions, a variable x introduced by a formula like $\Box A$ is "universal" in that an instantiation of x on one branch need not affect the value of x on other branches, thereby localising the effects of a variable instantiation to one branch. The technique entails creating and instantiating local duplicates of labelled formulae instead of the originals.

Finite diamond-rule: Applying the diamond-rule to $\Diamond A$ usually creates a new label. By using (a Gödelisation of) the formula A itself as the label instead, we guarantee that only a finite number of different labels (of a certain length) are used in the proof. In particular, different (identically labelled) occurrences of $\Diamond A$ generate the same unique label.

The paper is structured as follows: In Sections 2 and 3 we introduce the syntax and semantics of labelled modal tableaux. In Section 4 we introduce our

calculus and present an example; we prove its soundness and completeness in Sections 5 and 6, respectively. In Section 7 we describe our implementation and present experimental results; and in Section 8 we present our conclusions and discuss future work.

2 Syntax

The formulae of modal logics are built in the usual way from a denumerable non-empty set $\mathcal{P}$ of primitive propositions, the classical connectives $\wedge$ (conjunction), $\vee$ (disjunction), $\neg$ (negation), $\rightarrow$ (implication), and the non-classical unary modal connectives $\Box$ ("box") and $\Diamond$ ("diamond").

To reduce the number of tableau rules and the number of case distinctions in proofs, we restrict all considerations to implication-free formulae in negated normal form (NNF); thus negation signs appear in front of primitive propositions only. Using NNF formulae is no real restriction since *every* formula can be transformed into an equivalent NNF formula in linear time.

Labels are built from natural numbers and variables, with variables intended to capture the similarities between the $\forall$ quantifier of first-order logic and the $\Box$ modality of propositional modal logic. However, whereas first-order logic forbids an empty domain, the $\Box$ modality tolerates possible worlds with no successors.[1] To capture this (new) behaviour, variable positions in labels are made "conditional" on the existence of an appropriate successor by enclosing these conditional positions in parentheses.

Definition 1 *Let Vars be a set of variables, and let* $\mathbf{N}$ *be the set of natural numbers. Let* x, y, z *range over arbitrary members of Vars, let* n *and* m *range over arbitrary members of* $\mathbf{N}$*, and let* l *range over arbitrary members of* $Vars \cup \mathbf{N}$*. Then, the string* 1 *is a* ***label****; and if* σ *is a label, then so are* $\sigma.m$ *and* $\sigma.(l)$*. The* ***length*** *of a label* σ *is the number of dots it contains plus one, and is denoted by* $|\sigma|$*. The constituents of a label* σ *are called* ***positions*** *in* σ *and terms like "the* 1*st position" or "the* n*-th position" are defined in the obvious way. A position is* ***conditional*** *if it is of the form* (l)*, and a label is conditional if it contains a conditional position. By* $ipr(\sigma)$ *we mean the set of all non-empty* ***initial prefixes*** *of a label* σ*, excluding* σ *itself. A label is* ***ground*** *if it consists of (possibly conditional) members of* $\mathbf{N}$ *only. Let* $\mathcal{L}$ *be the set of all ground labels.*

When dealing with ground labels, we often do not differentiate between the labels $\sigma.n$ and $\sigma.(n)$, and we use $\sigma.[n]$ to denote that the label may be of either form. Note also that $\sigma.x$ (parentheses around x omitted) is not a label: the parentheses mark the positions that contain variables, or that used to contain variables before a substitution was applied.

Definition 2 *A set* Γ *of labels is* ***strongly generated*** *if:*

1. *there is some (root) label* $\rho \in \Gamma$ *such that* $\rho \in ipr(\sigma)$ *for all* $\sigma \in \Gamma \setminus \{\rho\}$*; and*

[1] To that extent, modal logics are similar to free logic, i.e., first-order logic where the domains of models may be empty [3].

2. $\sigma \in \Gamma$ *implies* $\tau \in \Gamma$ *for all* $\tau \in ipr(\sigma)$.

Since we deal with mono-modal logics with semantics in terms of rooted frames (see Section 3), we always assume that our labels form a strongly generated set with root $\rho = 1$. In any case, our definition of labels guarantees that all our labels begin with 1, and it is easy to see that the labels that appear in any of our tableaux are strongly generated.

Definition 3 *A* ***labelled tableau formula*** *(or just tableau formula) is a structure of the form* $X : \Delta : \sigma : A$*, where* X *is a subset of* $Vars \cup \mathbf{N}$*,* Δ *is a set of labels,* σ *is a label, and* A *is a formula in NNF. If the set* Δ *is empty, we use* $X : \sigma : A$ *as an abbreviation for* $X : \emptyset : \sigma : A$*. A tableau formula* $X : \Delta : \sigma : A$ *is* ***ground****, if* σ *and all labels in* Δ *are ground. If* $\mathcal{F}$ *is a set of labelled tableau formulae, then* $lab(\mathcal{F})$ *is the set* $\{\sigma \mid X : \Delta : \sigma : A \in \mathcal{F}\}$.

The intuitions behind the different parts of our "tableau formulae" are as follows: The fourth part A is just a traditional modal formula. The third part σ is a label, possibly containing variables introduced by the reduction of $\Box$ modalities. If the label σ is ground, then it corresponds to a particular path in the intended rooted tree model; for example, the ground label 1.1.1 typically represents the leftmost child of the leftmost child of the root 1. If σ contains variables, then it represents all the different paths (successors) that can be obtained by different instantiations of the variables, thereby capturing the semantics of the $\Box$ modalities that introduced them. Our rule for splitting disjunctions allows us to retain these variables in the labels of the two disjuncts, but because $\Box$ does not distribute over $\vee$, such variables then lose their "universal" force, meaning that these "free" variables can be instantiated only *once* in a tableau proof. We use the first component X to record the variables in the tableau formula ϕ that are "universal", meaning that ϕ can be used multiply in the same proof with different instantiations for these variables. The free variables in ϕ (that do not appear in X) can be used with only one instantiation since they have been pushed through the scope of an $\vee$ connective. The second part Δ, which can be empty, has a significance only if our calculus is applied to one of the four logics **KB**, **K5**, **KB4**, and **K45** (that are non-serial, but are symmetric or euclidean, see Section 3). It is empty for the other logics. The intuition of Δ is that the formula A has to be true in the possible world called σ only if the labels in Δ name legitimate worlds in the model under consideration. This feature has to be used, if (a) rule applications may shorten labels, which is the case if the logic is symmetric or euclidean, and (b) the logic is non-serial and, thus, the existence of successor worlds is not guaranteed. The set Δ can contain both universal and free variables, and some of them may appear in σ.

Definition 4 *Given a tableau formula* $\phi = X : \Delta : \sigma : A$*,* $Univ(\phi) = X$ *is the set of* ***universal variables*** *of* ϕ*, while* $Free(\phi) = \{x \text{ appears in } \sigma \text{ or } \Delta \mid x \notin X\}$ *is the set of* ***free variables*** *of* ϕ*. These notions are extended in the obvious way to obtain the sets* $Free(\mathcal{T})$ *and* $Univ(\mathcal{T})$ *of free and universal variables of a given tableau* $\mathcal{T}$ *(see Def. 5).*

Definition 5 *A **tableau** is a (finite) binary tree whose nodes are tableau formulae. A **branch** in a tableau $\mathcal{T}$ is a maximal path in $\mathcal{T}$.*[2] *A branch may be marked as being **closed**. If it is not marked as being closed, it is **open**. A tableau branch is **ground** if every formula on it is ground, and a tableau is ground if all its branches are ground.*

Since we deal with propositional modal logics, notions from first-order logic like variables and substitutions are needed only for handling semantic notions like the accessibility relation between worlds. Specifically, whereas substitutions in first-order logic assign terms to variables, here they assign numbers or other variables (denoting possibles worlds) to variables.

Definition 6 *A **substitution** is a (partial) function $\mu : Vars \to \mathbf{N} \cup Vars$. Substitutions are extended to labels and formulae in the obvious way. A substitution is **grounding** if its domain is the (whole) set Vars and its range is $\mathbf{N}$; that is, if it maps all variables in Vars to natural numbers. A substitution is a **variable renaming** if its range is Vars, and it replaces distinct variables by other distinct variables only. The **restriction** of a substitution μ to a set X of variables is denoted by $\mu_{|X}$.*

Definition 7 *Given a tableau $\mathcal{T}$ containing a tableau formula $X : \Delta : \sigma : A$, a tableau formula $X' : \Delta' : \sigma' : A$ is a $\mathcal{T}$**-renaming** of $X : \Delta : \sigma : A$ if there is a variable renaming μ such that $X' : \Delta' : \sigma' : A = (X : \Delta : \sigma : A)\mu$, and every variable introduced by μ is new to the tableau $\mathcal{T}$.*

3 Semantics

In this section we first introduce the Kripke semantics for modal logics, and then extend these semantics to labelled tableau formulae and tableau.

Definition 8 *A Kripke **frame** is a pair $\langle W, R \rangle$, where W is a non-empty set (of possible worlds) and R is a binary relation on W. A Kripke **model** is a triple $\langle W, R, V \rangle$, where the valuation V is a mapping from primitive propositions to sets of worlds. Thus, $V(p)$ is the set of worlds at which p is "true" under the valuation V. We write wRw' iff $(w, w') \in R$, and we say that world w' is **reachable** from world w, and that w' is a **successor** of w. A world $w \in W$ is **idealisable** if it has a successor in W.*

Definition 9 *Given some model $\langle W, R, V \rangle$, and some $w \in W$, we write $w \models p$ iff $w \in V(p)$. This satisfaction relation $\models$ is then extended to more complex formulae as usual. We say that w **satisfies** a formula A iff $w \models A$. A formula A is **valid** in a model $\langle W, R, V \rangle$, written as $\langle W, R, V \rangle \models A$, iff every world in W satisfies A. A formula A is **valid** in a frame $\langle W, R \rangle$, iff it is valid in every model $\langle W, R, V \rangle$ based on that frame. An axiom A is **valid** in a frame $\langle W, R \rangle$, iff every formula instance of it is valid in $\langle W, R \rangle$.*

[2] Where no confusion can arise, we identify a tableau branch with the set of tableau formulae it contains.

Name	Axiom	Property
(K)	$\Box(A \to B) \to (\Box A \to \Box B)$	—
(T)	$\Box A \to A$	reflexive
(D)	$\Box A \to \Diamond A$	serial
(4)	$\Box A \to \Box\Box A$	transitive
(5)	$\Diamond A \to \Box\Diamond A$	euclidean
(B)	$A \to \Box\Diamond A$	symmetric

Table 1: Basic axioms and their corresponding restrictions on the reachability relation.

The first two columns of Table 2 show the axiomatisations of the 15 basic logics that can be formed from the axioms shown in Table 1.

Definition 10 *Given one of the logics* **L** *listed in Table 2, a frame* $\langle W, R\rangle$ *is an* **L***-frame if each axiom of* **L** *is valid in* $\langle W, R\rangle$*. A model* $\langle W, R, V\rangle$ *is an* **L***-model if* $\langle W, R\rangle$ *is an* **L***-frame.*

It is well-known that the axioms listed in Table 1 are characterised by the properties of R listed next to them; see [10] for details. Thus, all **KT**-frames will have a reflexive accessibility relation R, and if a frame has a reflexive accessibility relation then it will validate axiom (T). Therefore, we associate these properties with logics as well, and say, for example, that a logic **L** is serial if all **L**-frames have a serial accessibility relation. Some care is needed here: for example the axiom (D) is not an axiom of **KT**, but it is valid in all **KT**-frames since it is implied by (T). Consequently the reachability relation R of all **KT**-models *is* serial.

As we shall soon see, ground labels capture a basic reachability relation between the worlds they name, where the world named by $\sigma.[n]$ is reachable from the world named by σ. A set of strongly generated ground labels can be viewed as a tree with root ρ, where $\sigma.[n]$ is an immediate child of σ (hence the name "strongly generated"). We formalise this as follows.

Definition 11 *Given a logic* **L** *and a set* Γ *of strongly generated ground labels with root* $\rho = 1$*, a label* $\tau \in \Gamma$ *is* **L-*accessible*** *from a label* $\sigma \in \Gamma$*, written as* $\sigma \rhd \tau$*, if the conditions set out in Table 2 are satisfied. A label* $\sigma \in \Gamma$ *is an* **L-*deadend***, *if no* $\tau \in \Gamma$ *is* **L***-accessible from* σ*.*

The following lemma shows that the **L**-accessibility relation $\rhd$ on labels captures the reachability relation R of **L**-frames exactly; see [10] for a proof. In particular, $\rhd$ has the properties like reflexivity, transitivity, etc. that are appropriate for the axioms of **L** (see Table 1).

Lemma 12 *If* Γ *is a strongly generated set of ground labels with root* $\rho = 1$*, then* $\langle \Gamma, \rhd\rangle$ *is an* **L***-frame.*

The traditional notion of satisfaction relates a world in a model with a formula or a set of formulae. When formulae are annotated with ground labels, the notion

Logic	Axioms	$\sigma \rhd \tau$	Logic	Axioms	$\sigma \rhd \tau$
K	(K)	$\tau = \sigma.[n]$	**KT**	(KT)	$\tau = \sigma.[n]$ or $\tau = \sigma$
KB	(KB)	$\tau = \sigma.[n]$ or $\sigma = \tau.[m]$	**K4**	(K4)	$\tau = \sigma.\theta$
K5	(K5)	$\tau = \sigma.[n]$, or $\lvert\sigma\rvert \geq 2$, $\lvert\tau\rvert \geq 2$	**K45**	(K45)	$\tau = \sigma.\theta$, or $\lvert\sigma\rvert \geq 2$, $\lvert\tau\rvert \geq 2$
KD	(KD)	**K**-condition, or σ is a **K**-deadend and $\sigma = \tau$	**KDB**	(KDB)	**KB**-condition, or $\lvert\Gamma\rvert = 1$ and $\sigma = \tau = 1$
KD4	(KD4)	**K4**-condition, or σ is a **K**-deadend and $\sigma = \tau$	**KD5**	(DK5)	**K5**-condition, or $\lvert\Gamma\rvert = 1$ and $\sigma = \tau = 1$
KD45	(KD45)	**K45**-cond., or $\lvert\Gamma\rvert = 1$, $\sigma = \tau = 1$	**KB4**	(KB4)	$\lvert\Gamma\rvert \geq 2$
B	(KTB)	$\tau = \sigma$, or $\tau = \sigma.[n]$, or $\sigma = \tau.[m]$	**S4**	(KT4)	$\tau = \sigma.\theta$ or $\tau = \sigma$
S5	(KT5)	for all σ, τ			

Table 2: Basic logics, axiomatic characterisations, and **L**-accessibility $\rhd$.

of satisfaction must be extended by a further "interpretation function" that maps ground labels to worlds; see [7, 10]. If the labels are allowed to contain free variables, and in particular, universal variables, then the notion of satisfaction must also allow for all possible instantiations of the universal variables, thus catering for many different "interpretation functions". The goal, as usual, is to define the notion of satisfiability so that our tableau expansion rules preserve this notion, and such that a "closed tableau" is not satisfiable.

We proceed incrementally by defining satisfiability for: ground labels; ground tableau formulae; non-ground tableau formulae; and finally for whole tableaux. But first we enrich models by the "interpretation function" that maps labels to worlds. Note that such interpretations give a meaning to *all* ground labels, not just to those that appear in a particular tableau.

Definition 13 *An* **L**-*interpretation is a pair* $\langle \mathbf{M}, \mathbf{I} \rangle$, *where* $\mathbf{M} = \langle W, R, V \rangle$, *is a Kripke* **L**-*model and* $\mathbf{I}$ *is a function* $\mathbf{I} : \mathcal{L} \to W \cup \{\bot\}$ *interpreting ground labels such that:*

(i) $\mathbf{I}(1) \in W$;

(ii) $\mathbf{I}(\sigma.(n)) = \mathbf{I}(\sigma.n)$ *for all* $\sigma.n$ *and* $\sigma.(n)$ *in* $\mathcal{L}$;

(iii) *for all* $\sigma \in \mathcal{L}$, *if* $\mathbf{I}(\tau) = \bot$ *for some* $\tau \in ipr(\sigma)$ *then* $\mathbf{I}(\sigma) = \bot$;

(iv) *if* $\sigma \rhd \tau$, $\mathbf{I}(\sigma) \in W$, $\mathbf{I}(\tau) \in W$, *and* $\mathbf{I}(\sigma)$ *is idealisable, then* $\mathbf{I}(\sigma)\, R\, \mathbf{I}(\tau)$.

Definition 14 *An* **L***-interpretation* $\langle \mathbf{M}, \mathbf{I} \rangle$*, where* $\mathbf{M} = \langle W, R, V \rangle$*,* ***satisfies*** *a ground label* σ*, if for all labels* $\tau.n \in ipr(\sigma) \cup \{\sigma\}$ *(that end in an unconditional label position):* $\mathbf{I}(\tau) \in W$ *implies* $\mathbf{I}(\tau.n) \in W$*. The* **L***-interpretation* $\langle \mathbf{M}, \mathbf{I} \rangle$ ***satisfies*** *a ground tableau formula* $X : \Delta : \sigma : A$*, if*

(a) $\mathbf{I}(\sigma) = \bot$*, or* $\mathbf{I}(\tau) = \bot$ *for some* $\tau \in \Delta$*, or* $\mathbf{I}(\sigma) \models A$*;*

(b) *if* $\mathbf{I}(\tau) \in W$ *for all* $\tau \in \Delta$*, then* $\langle \mathbf{M}, \mathbf{I} \rangle$ *satisfies* σ*.*

Thus, a tableau formula is satisfied by default if its label σ is undefined (that is, if $\mathbf{I}(\sigma) = \bot$) or if one of the labels in Δ is undefined. But because we deal only with strongly generated sets of labels with root 1, the twin requirements that every **L**-interpretation $\langle \mathbf{M}, \mathbf{I} \rangle$ define the label 1, and condition (b) in the above definition force the interpretation function $\mathbf{I}$ to "define" as many members of $\mathrm{ipr}(\sigma)$ as is possible. However, for a conditional ground label of the form $\tau.(n)$, where n is parenthesised, it is perfectly acceptable to have $\mathbf{I}(\tau.(n)) = \bot$ even if $\mathbf{I}(\tau) \in W$.

Example 15 *If* $\langle \mathbf{M}, \mathbf{I} \rangle$ *satisfies* $\sigma = 1.1.1$*, then* $\mathbf{I}(1)$*,* $\mathbf{I}(1.1)$*, and* $\mathbf{I}(1.1.1)$ *must be defined. If* $\sigma = 1.(1).1$*, then* $\mathbf{I}(1.(1))$ *need not be defined; but if it is, then* $\mathbf{I}(1.(1).1)$ *must be defined.*

The domain of every interpretation function $\mathbf{I}$ is the set of all *ground* labels $\mathcal{L}$, but our tableaux contain labels with variables. We therefore introduce a definition of satisfiability for non-ground tableau formulae capturing our intuitions that a label $\sigma.(x)$ stands for *all* possible successors of the label σ, and taking into account the special nature of universal variables.

Definition 16 *Given an* **L***-interpretation* $\langle \mathbf{M}, \mathbf{I} \rangle$ *and a grounding substitution* μ*, a (non-ground) tableau formula* ϕ *is* ***satisfied*** *by* $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$*, written as* $\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi$*, if for all grounding substitutions* λ*, the ground formula* $\phi\lambda_{|X}\mu$ *is satisfied by* $\langle \mathbf{M}, \mathbf{I} \rangle$ *(Def. 14). A set* $\mathcal{F}$ *of tableau formulae is satisfied by* $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$*, if every member of* $\mathcal{F}$ *is simultaneously satisfied by* $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$*.*

In the above definition, a ground formula $\phi\lambda_{|X}\mu$ is constructed from ϕ in two steps, such that the definition of satisfiablity for ground formulae can be applied. To cater for the differences between the free variables and universal variables, we use two substitutions: a fixed substitution μ and an arbitrary substitution λ. The first step, applying $\lambda_{|X}$ to ϕ instantiates the universal variables $x \in X$. The second step, applying μ to $\phi\lambda_{|X}$, instantiates the free variables. Therefore, the instantiation of universal variables $x \in X$ is given by the arbitrary substitution λ, and the instantiation of free variables $x \notin X$ is given by the fixed substitution μ.

Note, that in the following definition of satisfiable tableaux, there has to be a single satisfying **L**-interpretation for *all* grounding substitutions μ.

Definition 17 *A tableau* $\mathcal{T}$ *is* **L***-satisfiable if there is an* **L***-interpretation* $\langle \mathbf{M}, \mathbf{I} \rangle$ *such that for* every *grounding substitution* μ *there is some* open *branch* $\mathcal{B}$ *in* $\mathcal{T}$ *with* $\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \mathcal{B}$*.*

4 The Calculus

4.1 Overview

We now present an overview of our calculus, highlighting its main principles.

Our calculus is a refutation method. That is, to prove that a formula A is a theorem of logic $\mathbf{L}$, we first convert its negation $\neg A$ into NNF obtaining a formula B, and then test if B is $\mathbf{L}$-unsatisfiable. To do so, we start with the initial tableau whose single node is $\emptyset : \emptyset : 1 : B$ and repeatedly apply the tableau expansion rules, the substitution rule, and the closure rule until a closed tableau has been constructed. Since our rules preserve $\mathbf{L}$-satisfiability of tableaux, a closed tableau indicates that B is indeed $\mathbf{L}$-unsatisfiable, and hence that its negation A is $\mathbf{L}$-valid. Since $\mathbf{L}$-frames characterise the logic $\mathbf{L}$ we then know that A is a theorem of logic $\mathbf{L}$. Constructing a tableau for $\emptyset : \emptyset : 1 : B$ can be seen as a search for an $\mathbf{L}$-model for B. Each branch is a partial definition of a possible $\mathbf{L}$-model, and different substitutions give different $\mathbf{L}$-models. Our tableau rules extend *one* particular branch using *one* particular formula, thus differing *crucially* from the systematic methods in [6, 10] where a rule extends *all* branches that pass through one particular formula.

Free variables are used in the labels so that when the box-rule is applied in a world, the actual *ground* label of the successor world does not have to be guessed. Instead, free variables can be instantiated immediatedly before a branch is closed to make that closure possible. Note, however, that one single instantiation of the free variables has to be found that allows us to close all branches of a tableau *simultaneously*, and that instantiating a free variable (in the wrong way) to close one branch, can make it impossible to close other branches.

Because a world may have no successor, variable positions in labels have to be conditional to preserve soundness for non-serial logics.

Every variable is introduced into a label by the reduction of a box-formula like $\Box A$. Such a variable x in a tableau formula ϕ on branch $\mathcal{B}$ is "universal" if a renaming $\phi' = \phi\{x := x'\}$ of ϕ could be added to $\mathcal{B}$ without generating additional branches. That is, the modified tableau would be no more difficult to close than the original. An easy way to generate the renaming is to repeat the rule applications that lead to the generation of ϕ, starting from the box-rule application that created x. Once the renaming ϕ' is present on $\mathcal{B}$, the variable x never has to be instantiated to close $\mathcal{B}$ because ϕ' could be used instead of ϕ, thus instantiating x' instead of x. However, if x occurs on two separate branches in the tableau, then x is not universal because repeating these rule applications would generate at least one additional branch. Since the only rule that causes branching is the disjunctive rule, the two separate occurrences of x must have been created by a disjunctive rule application. Therefore, an application of the disjunctive rule to a formula ψ causes the universal variables of ψ to become free variables. Thus, all free variables are a result of a disjunction within the scope of a $\Box$, corresponding to the fact that $\Box$ does not distribute over $\vee$.

When the disjunctive rule "frees" universal variables, additional copies of the box-formula that generated them are needed. However, these additional copies are not generated by the box-rule, but by the disjunctive rule itself.

Our diamond-rule does not introduce a *new* label $\sigma.n$, when it is applied to $X:\Delta:\sigma:\Diamond A$. Instead, each formula $\Diamond A$ is assigned its own unique label $\lceil A \rceil$ which is a Gödelisation of A itself. This rule is easier to implement than the traditional one; and it guarantees that the number of different labels (of a certain length) in a proof is finite, thus restricting the search space.

The box-rule for symmetric and euclidean logics can shorten labels. For example, the tableau formula $X':\Delta':1:A$ is derived from $X:\Delta:1.(1):\Box A$ if the logic is symmetric. The semantics for serial logics guarantee that all labels define worlds, but in non-serial logics, the label 1 may be defined even though 1.(1) is undefined. To ensure that the formula $X':\Delta':1:A$ or one of its descendants is used to close a branch only if the label 1.(1) *is* defined, the label 1.(1) is made part of Δ' (see Section 4.3). Such problems do not occur when rule applications always lengthen labels since τ has to be defined if $\tau.l$ is defined.

All expansion rules are sound and *invertible* (some denominator of each rule is **L**-satisfiable *iff* the numerator is **L**-satisfiable). Thus, unlike traditional modal tableau methods where the order of (their non-invertible) rule applications is crucial [6, 10], the order of rule application is *immaterial*.

The differences in the calculi for different logics **L** is mainly in the box-rule, with different denominators for different logics. In addition, a simpler version of the closure rule can be used if the logic is serial.

4.2 Tableau Expansion Rules

There are four expansion rules, one for each type of complex (non-literal) formula. If we wanted to avoid NNF we would have four formula classes (α, β, ν, π) a la Smullyan [6], and an extra rule for double negation. Since we assume that all our formulae are in NNF, we need just one representative for each of the four classes.

As usual, in each rule, the formula above the horizontal line is its *numerator* (the premiss) and the formula(e) below the horizontal line, possibly separated by vertical bars, are its *denominators* (the conclusions). All expansion rules (including the box-rule) are "destructive"; that is, once the (appropriate) rule has been applied to a formula occurrence to expand a branch, that formula occurrence is not used again to expand that branch. Note that we permit multiple occurrences of the same formula on the same branch.

Definition 18 *Given a tableau $\mathcal{T}$, a new tableau $\mathcal{T}'$ may be constructed from $\mathcal{T}$ by applying one of the* **L-*expansion rules*** *from Table 3 as follows: If the numerator of a rule occurs on a branch $\mathcal{B}$ in $\mathcal{T}$, then the branch $\mathcal{B}$ is extended by the addition of the denominators of that rule. For the disjunctive rule the branch splits and the formulae in the right and left denominator, respectively, are added to the two resulting sub-branches instead.*

The box-rule(s) shown in Table 3 require explanation. The form of the rule is determined by the index **L** in the accompanying table. But some of the denominators have side conditions that determine when they are applicable. For example, the constraint $\sigma_6 = 1.l_6$ means that (5) is part of the denominator only

$$\frac{X:\Delta:\sigma:A\wedge B}{\begin{array}{c}X:\Delta:\sigma:A\\ X':\Delta':\sigma':B\end{array}}$$

Conjunctive rule. $X':\Delta':\sigma':B$ is a $\mathcal{T}$-renaming of $X:\Delta:\sigma:B$.

$$\frac{X:\Delta:\sigma:A\vee B}{\begin{array}{c|c}\emptyset:\Delta_1:\sigma_1:A & \emptyset:\Delta_1:\sigma_1:B\\ X_2:\Delta_2:\sigma_2:A\vee B & X_3:\Delta_3:\sigma_3:A\vee B\end{array}}$$

Disjunctive rule. For $1\leq i\leq 3$, the $\psi_i = X_i:\Delta_i:\sigma_i:A\vee B$ are $\mathcal{T}$-renamings of $\psi = X:\Delta:\sigma:A\vee B$ (the X_i are pairwise disjoint). If $X=\emptyset$ then ψ_2,ψ_3 are omitted.

$$\frac{X:\Delta:\sigma:\Diamond A}{X:\Delta:\sigma.\lceil A\rceil:A}$$

Diamond-rule. $\lceil\cdot\rceil$ is an arbitrary but fixed bijection from the set of formulae to $\mathbf{N}$.

$$\frac{X:\Delta:\sigma:\Box A}{\begin{array}{cc}X\cup\{x\}:\Delta:\sigma.(x):A & \text{(K)}\\ X_1\cup\{x_1\}:\Delta_1:\sigma_1.(x_1):\Box A & (4)\\ X_2\cup\{x_2\}:\Delta_2:\sigma_2.(x_2):\Box A & (4^d)\\ X_3:\Delta_3\cup\{\sigma_3\}:\tau_3:\Box A & (4^r)\\ X_4:\Delta_4:\sigma_4:A & \text{(T)}\\ X_5:\Delta_5\cup\{\sigma_5\}:\tau_5:A & \text{(B)}\\ X_6:\Delta_6\cup\{\sigma_6\}:1:\Box\Box A & (5)\end{array}}$$

Box-rule. For $1\leq i\leq 6$, the formulae $X_i:\Delta_i:\sigma_i:\Box A$ are $\mathcal{T}$-renamings of $X:\Delta:\sigma:\Box A$. The variables $x,x_1,x_2\in$ Vars are new to $\mathcal{T}$. The sets $X\cup\{x\}$, $X_1\cup\{x_1\}$, $X_2\cup\{x_2\}$, X_3, X_4, X_5, and X_6 are pairwise disjoint. In addition, $\sigma_3=\tau_3.l_3$, $\sigma_5=\tau_5.l_5$, $\sigma_6=1.l_6$, and $|\sigma_2|\geq 2$. The form of the denominator depends on the logic $\mathbf{L}$, and is determined by including every denominator corresponding to the entry for $\mathbf{L}$ in the table below.

Logics	Box-rule denominator	Logics	Box-rule denominator
K, **D**	(K)	**K45**, **K45D**	(K), (4), (4^r)
T	(K), (T)	**K4B**,	(K), (B), (4), (4^r)
KB, **KDB**	(K), (B)	**B**	(K), (T), (B)
K4, **KD4**	(K), (4)	**S4**	(K), (T), (4)
K5, **KD5**	(K), (4^d), (4^r), (5)	**S5**	(K), (T), (4), (4^r)

Table 3: Tableau expansion rules.

when the numerator of the box-rule is of the form $X:\Delta:1.l_6:\Box A$. Similarly, the constraints $\sigma_3 = \tau_3.l_3$ and $\sigma_5 = \tau_5.l_5$ for the (4^r) and (B) denominators mean these rules can be used only for a numerator of the form $X:\Delta:\sigma:\Box A$ where $|\sigma| \geq 2$, thereby guaranteeing that the *strictly shorter* labels τ_3 and τ_5 that appear in the respective denominators are properly defined. Note that the (4^d) denominator is the restriction of the (4) denominator to the case where $|\sigma| \geq 2$. The table indicates that the rules for a logic **L** and its serial version **LD** are identical because these logics are distinguished by the form of our closure rule; see Definition 21. Various other ways to define the calculi for serial logics exist; see [10].

4.3 The Substitution Rule and the Closure Rule

By definition, the substitution rule allows us to apply *any* substitution at *any* time to a tableau. In practice, however, it makes sense to apply only "useful" substitutions; that is, those most general substitutions which allow to close a branch of the tableau.

Definition 19 ***Substitution rule:*** *Given a tableau $\mathcal{T}$, a new tableau $\mathcal{T}' = \mathcal{T}\sigma$ may be constructed from $\mathcal{T}$ by applying a substitution σ to $\mathcal{T}$ that instantiates free variables in $\mathcal{T}$ with other free variables or natural numbers.*

In tableaux for modal logics without free variables as well as in free-variable tableaux for first-order logic, a tableau branch is closed if it contains complementary literals since this immediately implies the existence of an inconsistency. Here, however, this is not always the case because the labels of the complementary literals may be conditional. For example, the (apparently contradictory) pair $\emptyset:1.(1):p$ and $\emptyset:1.(1):\neg p$ is not necessarily inconsistent since the world $\mathbf{I}(1.(1))$ may not exist in the chosen model. Before declaring this pair to be inconsistent, we therefore have to ensure that $\mathbf{I}(1.(1)) \neq \bot$ for all **L**-interpretations satisfying the tableau branch $\mathcal{B}$ that is to be closed. Fortunately, this knowledge can be deduced from other formulae on $\mathcal{B}$. Thus in our example, a formula like $\psi = X:1.1:A$ on $\mathcal{B}$ would "justify" the use of the literal pair $\emptyset:1.(1):p$ and $\emptyset:1.(1):\neg p$ for closing the branch $\mathcal{B}$ since any **L**-interpretation $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfying $\mathcal{B}$ has to satisfy ψ, and, thus, $\mathbf{I}(1.(1)) = \mathbf{I}(1.1) \neq \bot$ has to be a world in the chosen model **M**. The crucial point is that the label 1.1 of ψ is *unconditional* exactly in the *conditional* positions of $\emptyset:1.(1):p$ and $\emptyset:1.(1):\neg p$. These observations are now extended to the general case of arbitrary *ground* labels.

Definition 20 *A ground label σ with j-th position $[n_j]$ $(1 \leq j \leq |\sigma|)$ is* ***justified*** *on a branch $\mathcal{B}$ if there is some set $\mathcal{F} \subseteq \mathcal{B}$ of tableau formulae such that for every j:*

1. *some label in $lab(\mathcal{F})$ has (an unconditional but otherwise identical) j-th position n_j; and*
2. *for all $\tau \in lab(\mathcal{F})$: if $|\tau| \geq j$ then the j-th position in τ is n_j or (n_j).*

Definition 21 *Given a tableau $\mathcal{T}$ and a substitution $\rho : Univ(\mathcal{T}) \to \mathbf{N}$ that instantiates universal variables in $\mathcal{T}$ with natural numbers, the* ***L-closure rule***

allows to construct a new tableau $\mathcal{T}'$ from $\mathcal{T}$ by marking $\mathcal{B}$ in $\mathcal{T}$ as closed provided that:

1. *the branch $\mathcal{B}\rho$ of $\mathcal{T}\rho$ contains a pair $X:\Delta:\sigma:p$ and $X':\Delta':\sigma:\neg p$ of complementary literals; and*
2. *(a) the logic* **L** *is serial, or (b) all labels in $\{\sigma\} \cup \Delta \cup \Delta'$ are ground and justified on $\mathcal{B}\rho$.*

Note that the substitution ρ that instantiates universal variables is not actually applied to the tableau when the branch is closed; it only has to exist.

By definition, only complementary *literals* close tableau branches, but in theory, pairs of complementary *complex formulae* could be used as well.

4.4 Tableau Proofs

We now have all the ingredients we need to define the notion of a tableau proof.

Definition 22 *A sequence $\mathcal{T}^0, \ldots, \mathcal{T}^r$ of tableaux is an* **L-*proof*** *for the* **L***-unsatisfiability of a formula A if:*

1. *$\mathcal{T}^0$ consists of the single node $\emptyset:\emptyset:1:A$;*
2. *for $1 \leq m \leq r$, the tableau $\mathcal{T}^m$ is constructed from $\mathcal{T}^{m-1}$ by applying an* **L***-expansion rule (Def. 18), the substitution rule (Def. 19), or the* **L***-closure rule (Def. 21); and*
3. *all branches in $\mathcal{T}^r$ are marked as closed.*

Theorems 23 and 25 state soundness and completeness for our calculus with respect to the Kripke semantics for logic **L**; the proofs can be found in Sections 5 and 6.

Theorem 23 (Soundness) *Let A be a formula in NNF. If there is an* **L***-proof $\mathcal{T}^0, \ldots, \mathcal{T}^r$ for the* **L***-unsatisfiability of A (Def. 22), then A is* **L***-unsatisfiable.*

We prove completeness for the non-deterministic and unrestricted version of the calculus, and also for all tableau procedures based on this calculus that deterministically choose the next formula for expansion (in a *fair* way) and that only apply most general closing substitutions.

Definition 24 *Given an open tableau $\mathcal{T}$, a* tableau procedure Ψ *deterministically chooses an open branch $\mathcal{B}$ in $\mathcal{T}$ and a non-literal tableau formula ψ on $\mathcal{B}$ for expansion.*

The tableau procedure Ψ is fair if, in the (possibly infinite) tableau that is constructed using Ψ (where no substitution is applied and no branch is closed), every formula has been used for expansion of every branch on which it occurs.

Theorem 25 (Completeness) *Let Ψ be a fair tableau procedure, and let A be an* **L***-unsatisfiable formula in NNF. Then there is a (finite) tableau proof $\mathcal{T}^0, \ldots, \mathcal{T}^r$ for the* **L***-unsatisfiability of A, where $\mathcal{T}^i$ is constructed from $\mathcal{T}^{i-1}$ $(1 \leq i \leq r)$ by*

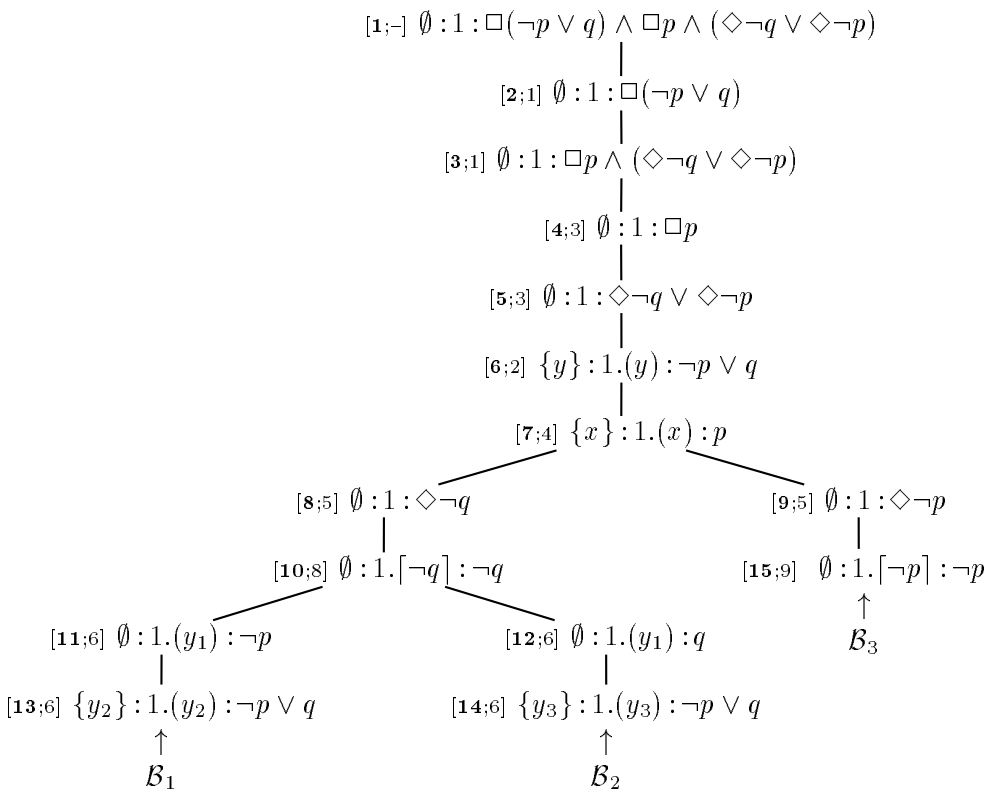


Figure 1: The tableau $\mathcal{T}$ from Example 26.

1. *applying the appropriate* **L***-expansion rule to the branch* $\mathcal{B}$ *and the formula* ψ *on* $\mathcal{B}$ *chosen by* Ψ *from* $\mathcal{T}^{i-1}$*; or*

2. *applying a most general substitution such that the* **L***-closure rule can be applied to a previously open branch in* $\mathcal{T}^{i-1}$*.*

Example 26 *We prove that* $A = \Box(p \to q) \to (\Box p \to (\Box q \wedge \Box p))$ *is a* **K***-theorem. To do this, we first transform the negation of* A *into NNF; the result is* $B = NNF(\neg A) = \Box(\neg p \vee q) \wedge \Box p \wedge (\Diamond \neg q \vee \Diamond \neg p)$*. The (fully expanded) tableau* $\mathcal{T}$*, that is part of the proof for the* **K***-unsatisfiability of* B *is shown in Figure 1. The nodes of the tableau are numbered; a pair* $[i; j]$ *is attached to the* i*-th node, the number* j *denotes that node* i *has been created by applying an expansion rule to the formula in node* j*. Note, that by applying the disjunctive rule to 6, the nodes 11 to 14 are added; 13 and 14 are renamings of 6. The variable* y_1 *is no longer universal in 11 and 12.*

When the substitution $\sigma = \{y_1/\lceil \neg q \rceil\}$ *is applied to* $\mathcal{T}$*, the branches of the resulting tableau* $\mathcal{T}\sigma$ *can be closed as follows, thereby completing the tableau proof: The left branch* $\mathcal{B}_1$ *of* $\mathcal{T}\sigma$ *can be closed by the universal variable substitution* $\rho_1 = \{x/\lceil \neg q \rceil\}$ *because* $\mathcal{B}_1\rho_1$ *then contains the complementary pair* $\{\lceil \neg q \rceil\} : 1.(\lceil \neg q \rceil) : p$ *and* $\emptyset : 1.(\lceil \neg q \rceil) : \neg p$ *in nodes 7 and 11, respectively. The label* $1.(\lceil \neg q \rceil)$ *of these literals is justified on* $\mathcal{B}_1\rho_1$ *by label* $1.\lceil \neg q \rceil$ *of formula 10. In this*

case, the complementary literals contain conditional labels which are only justified by a third formula on the branch, so checking for justification is indispensable. The middle branch $\mathcal{B}_2$ of $\mathcal{T}\sigma$ can be closed using the same universal variable substitution $\rho_2 = \rho_1 = \{x/\lceil\neg q\rceil\}$ as for the left branch. The branch $\mathcal{B}_2\rho_2$ then contains the complementary literals $\{\lceil\neg q\rceil\} : 1.(\lceil\neg q\rceil) : q$ and $\emptyset : 1.(\lceil\neg q\rceil) : \neg q$ in nodes 10 and 12. The label is again justified by formula 10, which in this case is one of the complementary literals. Note that the middle branch in $\mathcal{T}$ can be closed only by the substitution $\sigma = \{y_1/\lceil\neg q\rceil\}$, other choices will not suffice. The right branch $\mathcal{B}_3$ of $\mathcal{T}\sigma$ can be closed using the universal variable substitution $\rho_3 = \{x/\lceil\neg p\rceil\}$ as $\mathcal{B}_3\rho_3$ then contains the pair $\{\lceil\neg p\rceil\} : 1.(\lceil\neg p\rceil) : p$ and $\{\lceil\neg p\rceil\} : 1.\lceil\neg p\rceil : \neg p$ of complementary literals in nodes 7 resp. 15. The label $1.(\lceil\neg p\rceil)$ of node 7 is justified on $\mathcal{B}_3$ by formula 15.

The universal variable substitution $\rho_1 = \rho_2 = \{x/\lceil\neg q\rceil\}$ that closes $\mathcal{B}_1$ and $\mathcal{B}_2$ is incompatible with the substitution $\rho_3 = \{x/\lceil\neg p\rceil$ that closes $\mathcal{B}_3$. Therefore, if the variable x were not universal in formula 7, the tableau could not be closed; a second instance of formula 7 would have to be added.

5 Soundness

First, we introduce notation for the concatenation of substitutions:

Definition 27 *The **concatenation** $\mu \circ \lambda$ of substitutions μ and λ is defined by*

$$(\mu \circ \lambda)(x) = \mu(\lambda(x))$$

for all variables $x \in Vars$.

Note, that $O(\mu \circ \lambda) = O\lambda\mu$ for all objects O.

The following two lemmata, which will be used in the soundness proof, follow immediately from the definitions. The first one states, that a tableau formula ψ and a renaming ψ' of ψ are equivalent. The second lemma states that if a label σ is justified on a tableau branch $\mathcal{B}$ and $\mathcal{B}$ is satisfied by an interpretation $\langle \mathbf{M}, \mathbf{I}\rangle$, then $\mathbf{I}(\sigma)$ has to be a world in $\mathbf{M}$ (even if σ is conditional).

Lemma 28 *Let $\langle \mathbf{M}, \mathbf{I}\rangle$ be an $\mathbf{L}$-interpretation, μ a grounding substitution, ψ a formula in a tableau $\mathcal{T}$, and ψ' a $\mathcal{T}$-renaming of ψ. Then $\langle \mathbf{M}, \mathbf{I}, \mu\rangle \models \psi$ if and only if $\langle \mathbf{M}, \mathbf{I}, \mu\rangle \models \psi'$.*

Lemma 29 *Let $\langle \mathbf{M}, \mathbf{I}\rangle$ be an $\mathbf{L}$-interpretation, where $\mathbf{M} = \langle W, R, V\rangle$, let $\mathcal{B}$ be a tableau branch, and let σ be a ground label. If $\langle \mathbf{M}, \mathbf{I}, \mu\rangle$ $\mathbf{L}$-satisfies $\mathcal{B}$, and the label σ is justified on $\mathcal{B}$, then $\mathbf{I}(\sigma) \in W$.*

Definition 30 *An $\mathbf{L}$-interpretation $\langle \mathbf{M}, \mathbf{I}\rangle$, where $\mathbf{M} = \langle W, R, V\rangle$, is **canonical** provided that:*

1. *For all ground labels $\sigma = \tau.[n]$:*

$$\text{if } \mathbf{I}(\tau) \in W \text{ and } \mathbf{I}(\tau) \models \Diamond A_n, \text{ then } \mathbf{I}(\sigma) \models A_n,$$

where A_n is the formula for which $n = \lceil A_n\rceil$ ($\lceil\cdot\rceil$ is the bijection from the set of formulae to the set of natural numbers used for the diamond-rule).

2. If the logic **L** *is serial, then* $\mathbf{I}(\sigma) \in W$ *for all ground labels* σ.

The restriction to canonical interpretations only makes sense because every **L**-model **M** **L**-satisfying a formula A can be combined with a label interpretation **I**, such that $\langle \mathbf{M}, \mathbf{I} \rangle$ is canonical and **L**-satisfies the intial tableau $\emptyset : \emptyset : 1 : A$.

Lemma 31 *Given a formula* A *in NNF and an* **L***-model* **M** **L***-satisfying* A*, there is a canonical* **L***-interpretation* $\langle \mathbf{M}, \mathbf{I} \rangle$ *that* **L***-satisfies the tableau consisting of the singleton tableau formula* $\emptyset : \emptyset : 1 : A$.

Proof. Since $\mathbf{M} = \langle W, R, V \rangle$ **L**-satisfies A, we know that there is some world $w_1 \in W$ such that $w_1 \models A$. Now, for $n \geq 1$, let A_n be the formula for which $n = \lceil A_n \rceil$ (where $\lceil \cdot \rceil$ is the bijection from the set of formulae to the set of natural numbers used for the diamond-rule) and create **I** as follows: Put $\mathbf{I}(1) = w_1$, and for every ground label of the form $\tau.n$:

- if there is a world $w \in W$ such that $\mathbf{I}(\tau) R w$ and $w \models A_n$ then put $\mathbf{I}(\tau.n) = \mathbf{I}(\tau.(n)) = w$;
- else, if there is no such world w, but there is a world w' that is reachable from $\mathbf{I}(\tau)$, then put $\mathbf{I}(\tau.n) = \mathbf{I}(\tau.(n)) = w'$;
- else, if there is no world reachable from $\mathbf{I}(\tau)$, put $\mathbf{I}(\tau.n) = \mathbf{I}(\tau.(n)) = \bot$.

The **L**-interpretation $\langle \mathbf{M}, \mathbf{I} \rangle$ is canonical by way of its definition, and in addition **L**-satisfies the tableau consisting of $\phi_0 = \emptyset : \emptyset : 1 : A$. To prove this, we have to show that for all grounding substitutions μ there is a branch $\mathcal{B}$ in this tableau such that for all substitutions λ the ground formula $\phi_0 \lambda_{|\emptyset} \mu$ is **L**-satisfied by $\langle \mathbf{M}, \mathbf{I} \rangle$. Since $\phi_0 \lambda_{|\emptyset} \mu = \phi_0$ this reduces to showing that (a) $\mathbf{I}(1) = \bot$ or $\mathbf{I}(1) \models A$, and (b) $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies the label 1. Condition (a) is satisfied since $\mathbf{I}(1) \models A$ by choice, and Condition (b) holds because there are no labels $\tau.n$ in $\mathrm{ipr}(1) \cup \{1\}$. ∎

The following lemmata state that **L**-satisfiability by canonical interpretations is preserved if an expansion rule (Lemma 32), the substitution rule (Lemma 33), or the closure rule (Lemma 34) is applied to a tableau.

Lemma 32 *If the tableau* $\mathcal{T}$ *is* **L***-satisfied by the canonical* **L***-interpretation* $\langle \mathbf{M}, \mathbf{I} \rangle$*, and* $\mathcal{T}'$ *is constructed from* $\mathcal{T}$ *by applying an* **L***-expansion rule, then* $\langle \mathbf{M}, \mathbf{I} \rangle$ **L***-satisfies* $\mathcal{T}'$ *as well.*

Proof. We show that for each grounding substitution μ, there is a branch $\mathcal{B}'$ in $\mathcal{T}'$ that is **L**-satisfied by $\langle \mathbf{M}, \mathbf{I} \rangle$.

By assumption, $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies some branch $\mathcal{B}$ of $\mathcal{T}$. If $\mathcal{T}'$ is constructed from $\mathcal{T}$ by expanding a branch different from $\mathcal{B}$, then $\mathcal{B}$ is a branch of $\mathcal{T}'$ as well, and we are through. For the case that $\mathcal{T}'$ is constructed from $\mathcal{T}$ by expanding the branch $\mathcal{B}$, we show that $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies one of the branches of $\mathcal{T}'$ by cases according to which expansion rule is applied.

Conjunctive rule: Let $\phi = X : \Delta : \sigma : (A \wedge B)$ be the formula on $\mathcal{B}$, such that $\mathcal{T}'$ is constructed from $\mathcal{T}$ by adding $\phi_1 = X : \Delta : \sigma : A$ and $\phi_2' = X' : \Delta' : \sigma' : B$ to $\mathcal{B}$.

$\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi$ implies that for all grounding substitutions λ the following holds (where $\varsigma = \sigma \lambda_{|X} \mu$):

1. (a) $\mathbf{I}(\varsigma) = \bot$; or (b) $\mathbf{I}(\xi) = \bot$ for some $\xi \in \Delta\lambda_{|X}\mu$; or (c) $\mathbf{I}(\varsigma) \models A \wedge B$, and thus $\mathbf{I}(\varsigma) \models A$ and $\mathbf{I}(\varsigma) \models B$.

2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies ς.

This implies that $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies $\phi_1 = X : \Delta : \sigma : A$ and $\phi_2 = X : \Delta : \sigma : B$ and thus, according to Lemma 28, the renaming ϕ_2'; therefore, $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies the branch $\mathcal{B} \cup \{\phi_1, \phi_2'\}$ in $\mathcal{T}'$.

Disjunctive rule: Let $\phi = X : \Delta : \sigma : (A \vee B)$ be the formula in $\mathcal{B}$, such that $\mathcal{T}'$ is constructed from $\mathcal{T}$ by adding $\phi_1' = \emptyset : \Delta_1 : \sigma_1 : A$ and $\phi' = X_2 : \sigma_2 : \Delta_2 : A \vee B$ to $\mathcal{B}$ obtaining $\mathcal{B}_1'$ and adding $\phi_2' = \emptyset : \Delta_1 : \sigma_1 : B$ and $\phi'' = X_3 : \Delta_3 : \sigma_3 : A \vee B$ to $\mathcal{B}$ obtaining $\mathcal{B}_2'$.

$\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi$ implies that both ϕ' and ϕ'' are **L**-satisfied by $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ (using Lemma 28), and it implies that for all grounding substitutions λ' the following holds (where $\varsigma = \sigma\lambda'_{|X}\mu$):

1. (a) $\mathbf{I}(\varsigma) = \bot$, or (b) $\mathbf{I}(\xi) = \bot$ for some $\xi \in \Delta\lambda'_{|X}\mu$, or (c) $\mathbf{I}(\varsigma) \models A \vee B$, which implies that $\mathbf{I}(\varsigma) \models A$ or $\mathbf{I}(\varsigma) \models B$.

2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda'_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies ς.

This is in particular true if λ' is chosen to be equal to μ, and thus for the label $\varsigma = \sigma\mu_{|X}\mu = \sigma\mu = \sigma\lambda_{|\emptyset}\mu$ and the set $\Delta\lambda'_{|X}\mu = \Delta\lambda_{|\emptyset}\mu$, where λ is an arbitrary grounding substitution.

This implies that $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies $\phi_1 = \emptyset : \Delta : \sigma : A$ or it **L**-satisfies $\phi_2 = \emptyset : \Delta : \sigma : B$ and thus, according to Lemma 28, (at least) one of the renameings ϕ_1' and ϕ_2'; therefore, $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies one of the branches $\mathcal{B}_1'$ and $\mathcal{B}_2'$ in $\mathcal{T}'$.

Diamond-rule: Let $\phi = X : \Delta : \sigma : \Diamond A$ be the formula on $\mathcal{B}$, such that $\mathcal{T}'$ is constructed from $\mathcal{T}$ by adding $\phi_1 = X : \Delta : \sigma.\lceil A \rceil : A$ to $\mathcal{B}$.

$\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi$ implies that for all grounding substitutions λ the following holds (where $\varsigma = \sigma\lambda_{|X}\mu$):

1. (a) $\mathbf{I}(\varsigma) = \bot$ which implies $\mathbf{I}(\varsigma.\lceil A \rceil) = \bot$, or (b) $\mathbf{I}(\xi) = \bot$ for some $\xi \in \Delta\lambda_{|X}\mu$, or (c) $\mathbf{I}(\varsigma) \models \Diamond A$ which implies that there is a world $w \in W$ that is reachable from $\mathbf{I}(\varsigma)$, such that $w \models A$; thus, because $\langle \mathbf{M}, \mathbf{I} \rangle$ is canonical, $\mathbf{I}(\varsigma.\lceil A \rceil) \in W$ and $\mathbf{I}(\varsigma.\lceil A \rceil) \models A$.

2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies ς; in that case, (a) if $\mathbf{I}(\varsigma) = \bot$, then $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies $\varsigma.\lceil A \rceil$ as well, because ς has to be conditional, (b) if $\mathbf{I}(\varsigma) \neq \bot$, then $\mathbf{I}(\varsigma) \models \Diamond A$, which implies by definition of canonical interpretations that $\mathbf{I}(\varsigma.\lceil A \rceil) \in W$.

This implies that $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies $\phi_1 = X : \Delta : \sigma.\lceil A \rceil : A$ and, therefore, **L**-satisfies the branch $\mathcal{B} \cup \{\phi_1\}$ in $\mathcal{T}'$.

Box-rule: Let $\phi = X : \Delta : \sigma : \Box A$ be the formula in $\mathcal{B}$, such that $\mathcal{T}'$ is constructed from $\mathcal{T}$ by adding to $\mathcal{B}$—according to Table 3—the formulae

$$\phi_{(\mathrm{K})} \quad = \quad X \cup \{x\} : \Delta : \sigma.(x) : A$$

$$\begin{aligned}
\phi'_{(4)} = \phi'_{(4^d)} &= X_1 \cup \{x_1\} : \Delta_1 : \sigma_1.(x_1) : \Box A \\
\phi'_{(4^r)} &= X_3 : \Delta_3 \cup \{\sigma_3\} : \tau_3 : \Box A \\
\phi'_{(\mathrm{T})} &= X_4 : \Delta_4 : \sigma_4 : A \\
\phi'_{(\mathrm{B})} &= X_5 : \Delta_5 \cup \{\sigma_5\} : \tau_4 : A \\
\phi'_{(5)} &= X_6 : \Delta_6 \cup \{\sigma_6\} : 1 : \Box\Box A
\end{aligned}$$

$\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi$ implies that for all grounding substitutions λ the following holds (where $\varsigma = \sigma\lambda_{|X}\mu$):

1. (a) $\mathbf{I}(\varsigma) = \bot$, or (b) $\mathbf{I}(\xi) = \bot$ for some $\xi \in \Delta\lambda_{|X}\mu$, or (c) $\mathbf{I}(\varsigma) \models \Box A$.
2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies ς.

- We show that $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ $\mathbf{L}$-satisfies $\phi_{(\mathrm{K})} = X \cup \{x\} : \Delta : \sigma.(x) : A$. For all grounding substitutions λ the following holds (where $n = \mu(x)$, and thus $\sigma.(x)\lambda_{|X\cup\{x\}}\mu = \varsigma.(n)$):
  1. If $\mathbf{I}(\varsigma.(n)) \neq \bot$ then $\mathbf{I}(\varsigma) \neq \bot$ and $\mathbf{I}(\varsigma.(n))$ is reachable from $\mathbf{I}(\varsigma)$; if, in addition, $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu = \Delta\lambda_{|X\cup\{x\}}\mu$, then we can conclude that $\mathbf{I}(\varsigma) \models \Box A$. By the definition of the box operator, this implies $\mathbf{I}(\varsigma.(n)) \models A$.
  2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ satisfies the label $\varsigma.(n)$ because it satisfies ς and the extension (n) is conditional.

- $\phi'_{(4)}$ is only part of the numerator of the box-rule for (a) transitive logics (as (4) numerator of the box-rule) and (b) euclidean logics if $|\sigma| \geq 2$ (as (4^d) numerator of the box-rule). We show that if (a) the reachability relation R of $\mathbf{M}$ is transitive or (b) R is euclidean and $|\sigma| \geq 2$, then $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ $\mathbf{L}$-satisfies $\phi_{(4)} = X \cup \{x\} : \Delta : \sigma.(x) : \Box A$ and thus (using Lemma 28) $\mathbf{L}$-satisfies $\phi'_{(4)}$. For all grounding substitutions λ the following holds (where $n = \mu(x)$, i.e., $\sigma.(x)\lambda_{|X\cup\{x\}}\mu = \varsigma.(n)$):
  1. If $\mathbf{I}(\varsigma.(n)) = w' \neq \bot$ then $\mathbf{I}(\varsigma) = w \neq \bot$ and wRw'; if, in addition, $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu = \Delta\lambda_{|X\cup\{x\}}\mu$, then we can conclude that $w \models \Box A$. Suppose w' has no successors, then $w' \models \Box A$ for any A vacuously. If w' has a successor, then let w'' be any world such that $w'Rw''$. (a) If R is transitive, this immediately implies wRw''. (b) If R is euclidean and $|\varsigma| \geq 2$, then there is a world w_0 such that w_0Rw. We can derive wRw'' as follows: w_0Rw implies wRw, wRw' and wRw implies $w'Rw$, $w'Rw$ and $w'Rw''$ implies wRw''. Now, since $w \models \Box A$ we have $w'' \models A$, and since w'' is an arbitrary world reachable from w', $w' \models \Box A$.
  2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ satisfies the label $\varsigma.(n)$ because it satisfies ς and the extension (n) is conditional.

- $\phi'_{(4^r)}$ and $\phi'_{(5)}$ are only part of the numerator of the box-rule for euclidean logics $\mathbf{L}$. We show that if the reachability relation R of $\mathbf{M}$ is euclidean then

$\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies the tableau formula $\phi_{(4^r)} = X : \Delta \cup \{\sigma\} : \tau : \Box A$, where $\sigma = \tau.l$, and tableau formula $\phi_{(5)} = X : \Delta \cup \{\sigma\} : 1 : \Box\Box A$, where $\sigma = 1.l$, and thus (using Lemma 28) **L**-satisfies $\phi'_{(4^r)}$ and $\phi'_{(5)}$. For all grounding substitutions λ the following holds (where $\tau\lambda_{|X}\mu = \varsigma'$ and $[n] = \mu(l)$, i.e., $\varsigma = \varsigma'.[n]$):

1. If $\mathbf{I}(\varsigma) = \bot$ or there is some label ξ in $\Delta\lambda_{|X}\mu$ such that $\mathbf{I}(\xi) = \bot$, then there is a label ξ in $(\Delta \cup \{\sigma\})\lambda_{|X}\mu = \Delta\lambda_{|X}\mu \cup \{\varsigma\}$ such that $\mathbf{I}(\xi) = \bot$. Otherwise, if $\mathbf{I}(\varsigma) = w \models \Box A$, then $\mathbf{I}(\varsigma') = w' \neq \bot$ and $w'Rw$. Suppose w' has no successors, then $w' \models \Box A$ for any A vacuously. If w' has a successor, then let w'' be any world such that $w'Rw''$. Since the logic **L** is euclidean, $w'Rw$ and $w'Rw''$ implies wRw'' and $w''Rw$. Therefore, $w \models \Box A$ implies $w'' \models A$. Since this holds for all worlds w'' reachable from w', we can conclude $\mathbf{I}(\varsigma') = w' \models \Box A$.

   Moreover, if w''' is any world such that $w''Rw'''$, then $w''Rw$ implies wRw''' and thus $w''' \models A$. Since, again, w''' was chosen arbitrarily, this implies $w'' \models \Box A$ and $w' \models \Box\Box A$. This holds in particular for the case where $\sigma = 1.l$ and $w' = \mathbf{I}(1)$.
2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ satisfies the label $\varsigma' \in \mathrm{ipr}(\varsigma)$, because it satisfies ς.

- $\phi'_{(\mathrm{T})}$ is only part of the numerator of the box-rule for reflexive logics **L**. We show that if the reachability relation R of **M** is reflexive, then $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies $\phi_{(\mathrm{T})} = X : \Delta : \sigma : A$ and thus (using Lemma 28) **L**-satisfies $\phi'_{(\mathrm{T})}$. For all grounding substitutions λ the following holds: $\mathbf{I}(\varsigma) = \bot$, or $\mathbf{I}(\xi) = \bot$ for some $\xi \in \Delta\lambda_{|X}\mu$, or $\mathbf{I}(\varsigma) \models \Box A$ which implies $\mathbf{I}(\varsigma) \models A$ (because R is reflexive).

- $\phi'_{(\mathrm{B})}$ is only part of the numerator of the box-rule for symmetric logics **L**. We show that if the reachability relation R of **M** is symmetric then $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies $\phi_{(\mathrm{B})} = X : \Delta \cup \{\sigma\} : \tau : A$, where $\sigma = \tau.l$, and thus (using Lemma 28) **L**-satisfies $\phi'_{(\mathrm{B})}$. For all grounding substitutions λ the following holds (where $\varsigma' = \tau\lambda_{|X}\mu$ and $[n] = l\mu$):

  1. If $\mathbf{I}(\varsigma) = \bot$ or there is some label ξ in $\Delta\lambda_{|X}\mu$ such that $\mathbf{I}(\xi) = \bot$, then there is a label ξ in $(\Delta \cup \{\sigma\})\lambda_{|X}\mu = \Delta\lambda_{|X}\mu \cup \{\varsigma\}$ such that $\mathbf{I}(\xi) = \bot$. Otherwise, if $\mathbf{I}(\varsigma) = w \models \Box A$, then $\mathbf{I}(\varsigma') = w' \neq \bot$ and $w'Rw$ (since $\varsigma = \varsigma'.[n]$). Because R is symmetric this implies wRw' and thus $\mathbf{I}(\varsigma') = w' \models A$.
  2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta\lambda_{|X}\mu$, then $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ satisfies the label $\varsigma' \in \mathrm{ipr}(\varsigma)$, because it satisfies ς.

∎

Lemma 33 *If the tableau $\mathcal{T}$ is* **L***-satisfied by the canonical* **L***-interpretation $\langle \mathbf{M}, \mathbf{I} \rangle$, and $\mathcal{T}'$ is constructed from $\mathcal{T}$ by applying the substitution rule, then $\langle \mathbf{M}, \mathbf{I} \rangle$* **L***-satisfies $\mathcal{T}'$ as well.*

Proof. Let ρ be the substitution that has been applied to derive $\mathcal{T}'$ from $\mathcal{T}$. We have to show that for each grounding substitution μ, there is a branch $\mathcal{B}'$ in $\mathcal{T}'$ that is **L**-satisfied by $\langle \mathbf{M}, \mathbf{I} \rangle$.

Let μ be an arbitrary but fixed grounding substitution. By assumption, $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies some branch $\mathcal{B}$ of $\mathcal{T}$. Let $\phi = X : \Delta : \sigma : A$ be an arbitrary formula on $\mathcal{B}$.

Let λ be an arbitrary grounding substitution, and define the substitution λ' by $\lambda' = \lambda \circ \rho$. Then the substitutions $\mu \circ \lambda_{|X} \circ \rho$ and $\mu \circ \lambda'_{|X}$ are identical, because ρ does not instantiate variables in X, and thus $\lambda'_{|X} = (\lambda \circ \rho)_{|X} = \lambda_{|X} \circ \rho$.

Because $\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi$, the following holds (where $\varsigma' = \sigma \lambda'_{|X} \mu$):

1. (a) $\mathbf{I}(\varsigma') = \bot$, or (b) $\mathbf{I}(\xi) = \bot$ for some $\xi \in \Delta \lambda'_{|X} \mu$, or (c) $\mathbf{I}(\varsigma') \models A$.
2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta \lambda'_{|X} \mu$, then $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies ς'.

This implies, because $\mu \circ \lambda_{|X} \circ \rho = \mu \circ \lambda'_{|X}$, that the following holds (where $\varsigma = \sigma \rho \lambda_{|X} \mu$):

1. (a) $\mathbf{I}(\varsigma) = \bot$, or (b) $\mathbf{I}(\xi) = \bot$ for some $\xi \in \Delta \rho \lambda_{|X} \mu$, or (c) $\mathbf{I}(\varsigma) \models A$.
2. If $\mathbf{I}(\xi) \neq \bot$ for all $\xi \in \Delta \rho \lambda_{|X} \mu$, then $\langle \mathbf{M}, \mathbf{I} \rangle$ satisfies ς.

Thus, $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ **L**-satisfies $\phi\rho$. ∎

Lemma 34 *If the tableau $\mathcal{T}$ is* **L***-satisfied by the canonical* **L***-interpretation $\langle \mathbf{M}, \mathbf{I} \rangle$, and $\mathcal{T}'$ is constructed from $\mathcal{T}$ by applying the* **L***-closure rule, then $\langle \mathbf{M}, \mathbf{I} \rangle$* **L***-satisfies $\mathcal{T}'$ as well.*

Proof. $\mathcal{T}'$ is obtained from $\mathcal{T}$ by marking a branch $\mathcal{B}$ in $\mathcal{T}$ as being closed, because it contains formulae $\phi_1 = X_1 : \Delta_1 : \sigma_1 : p$ and $\phi_2 = X_2 : \Delta_2 : \sigma_2 : \neg p$, and there is a substitution ρ of the universal variables in $\mathcal{T}$ such that $\sigma_1 \rho_{|X_1} = \sigma_2 \rho_{|X_2} = \xi$ and (a) the logic **L** is serial, or (b) all labels ς in $\{\xi\} \cup \Delta_1 \rho_{|X_1} \cup \Delta_2 \rho_{|X_2}$ are ground and justified on $\mathcal{B}$. Suppose the branch $\mathcal{B}$ were **L**-satisfied by $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ for some grounding substitution μ. Then $\mathbf{I}(\varsigma\mu) \in W$, because

1. if the logic **L** is serial, then $\mathbf{I}(\varsigma\mu) \in W$ since $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ is canonical.
2. otherwise, ς is ground and justified, and thus $\varsigma\mu = \varsigma$ and $\mathbf{I}(\varsigma) \in W$ according to Lemma 29.

Therefore, $\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi_1$ implies $\mathbf{I}(\xi\mu) = \mathbf{I}(\sigma_1 \rho_{|X_1} \mu) \models p$, and $\langle \mathbf{M}, \mathbf{I}, \mu \rangle \models \phi_2$ implies $\mathbf{I}(\xi\mu) = \mathbf{I}(\sigma_2 \rho_{|X_2} \mu) \models \neg p$. This, however, is not possible.

Thus, our assumption is wrong, and $\mathcal{B}$ is not **L**-satisfied by $\langle \mathbf{M}, \mathbf{I}, \mu \rangle$ for any μ. But then there has to be a different branch $\mathcal{B}_0$ in $\mathcal{T}$ for all μ, that occurs in $\mathcal{T}'$ as well and is not affected by marking the branch $\mathcal{B}$ as being closed. ∎

Now we have everything at hand that is needed to prove soundness of our calculus:

Theorem 35 *Let A be a formula in NNF. If there is an* **L***-proof $\mathcal{T}^0, \ldots, \mathcal{T}^r$ for the* **L***-unsatisfiability of A (Def 22), then A is* **L***-unsatisfiable.*

Proof. For a contradiction, suppose there is an **L**-proof $\mathcal{T}^0, \ldots, \mathcal{T}^r$ for the **L**-unsatisfiability of A, but that A is **L**-satisfiable. Then there is an **L**-model **M** of A and by Lemma 31 there is a canonical **L**-interpretation of $\mathcal{T}^0$. Lemmata 32, 33 and 34 imply that **L**-satisfiability by canonical **L**-interpretations is preserved in tableau proofs. Hence the tableau $\mathcal{T}^r$ is **L**-satisfied by a canonical **L**-interpretation as well. But by definition of a tableau proof, all branches in $\mathcal{T}^r$ are marked as being closed, thus the tableau $\mathcal{T}^r$ cannot possibly be **L**-satisfiable. ∎

Note that if a tableau is **L**-satisfiable then it is **L**-satisfiable by a *canonical* interpretation. Thus, **L**-satisfiability is preserved in general; it is, however, quite difficult (if not impossible) to prove this directly without using the notion of canonical interpretations.

6 Completeness

We now turn to the completeness of our calculus. The completeness theorem can be stated in two contraposing ways: Let A be a formula in NNF:

> If A is **L**-unsatisfiable, then there is a tableau proof $\mathcal{T}^0, \ldots, \mathcal{T}^r$ for $\emptyset : \emptyset : 1 : A$.

or equivalently

> If there is no tableau proof for $\emptyset : \emptyset : 1 : A$ then A is **L**-satisfiable.

We first prove the completeness theorem as stated in the paper viz: Let Ψ be a fair tableau procedure, and let A be an **L**-unsatisfiable formula in NNF. Then there is a (finite) tableau proof $\mathcal{T}^0, \ldots, \mathcal{T}^r$ for the **L**-unsatisfiability of A, where $\mathcal{T}^i$ is constructed from $\mathcal{T}^{i-1}$ $(1 \leq i \leq r)$ by

- applying the appropriate **L**-expansion rule to the branch $\mathcal{B}$ and the formula ψ on $\mathcal{B}$ chosen by Ψ from $\mathcal{T}^{i-1}$; or
- applying a most general substitution such that the **L**-closure rule can be applied to a previously open branch in $\mathcal{T}^{i-1}$.

So suppose we are given a fair tableau procedure Ψ and an initial tableau $\emptyset : \emptyset : 1 : A$. We prove the theorem in a rather roundabout way following the method of Beckert and Posegga [2]:

Step 1 We define the notion of a ground **L**-Hintikka set (of tableau formulae) and show that every Hintikka set is **L**-satisfiable in some **L**-interpretation $\langle \mathbf{M}, \mathbf{I} \rangle$.

Step 2 We consider the sequence $(\mathcal{T}_n)_{n \geq 0}$ that results from our fair tableau procedure Ψ without closing branches or applying substitutions, and define the infinite tableau $\mathcal{T}_\infty$ to be the limit of the $\mathcal{T}_n$.

Step 3 Assuming that there is *no* substitution of the variables in $\mathcal{T}_\infty$ that gives a closed instance of $\mathcal{T}_\infty$, we define a particular substitution θ_∞ and show that $\mathcal{T}_\infty\theta_\infty$ contains at least one branch that forms a Hintikka set. Step 1 then gives us an **L**-satisfiable set, and in particular an **L**-model for the formula A in root $\emptyset:\emptyset:1:A$ of $\mathcal{T}_\infty\theta_\infty$. This allows us to prove a lemma stating that if there is no substitution that closes $\mathcal{T}_\infty$ then A is **L**-satisfiable.

Step 4 The contrapositive of this lemma is: If A is **L**-unsatisfiable, then there is at least one substitution θ that, when applied, allows to close all branches in $\mathcal{T}_\infty$. Thus there is an n such that all branches in the finite tableau $\mathcal{T}_n\theta$ can be closed. We construct another tableau $\mathcal{T}'_n$ from $\mathcal{T}_n$ by starting at the end of each branch and removing all formulae that are not needed to close the corresponding branch in $\mathcal{T}_n\theta$ (that includes justification). Note that $\mathcal{T}'_n\theta$ is still closed but is unlikely to be of uniform depth.

Step 5 We prove that if $\mathcal{T}'\theta$ is closed, then the substitution θ can be decomposed so that: $\theta = \theta'\circ\xi_r\circ\xi_{r-1}\circ...\circ\xi_1$ where ξ_i is a most general closing substitution for the instantiation $(\mathcal{B}_i)\xi_1\xi_2...\xi_{i-1}$ of the i-th branch $\mathcal{B}_i$ in $\mathcal{T}'$. And θ' is the part of θ that is not actually needed to close $\mathcal{T}'$.

Step 6 We prove the restriction to a fair tableau procedure and to using most general closing substitutions is complete.

6.1 Step 1: Labelled Hintikka Sets and L-satisfiability.

Definition 36 *A set $\mathcal{X}$ of ground labelled formulae is an* **L***-Hintikka set, if it satisfies the following conditions:*

1. *lab($\mathcal{X}$) is a strongly generated set of labels with root 1.*
2. *There is no primitive proposition p such that (a) both $X:\Delta_1:\sigma:p$ and $Y:\Delta_2:\sigma:\neg p$ are in $\mathcal{X}$, and (b) the logic* **L** *is serial or all labels in $\{\sigma\}\cup\Delta_1\cup\Delta_2$ are justified in $\mathcal{X}$.*
3. *If $X:\Delta:\sigma:A\wedge B\in\mathcal{X}$ then $X:\Delta:\sigma:A\in\mathcal{X}$ and $X:\Delta:\sigma:B\in\mathcal{X}$.*
4. *If $X:\Delta:\sigma:A\vee B\in\mathcal{X}$ then $X:\Delta:\sigma:A\in\mathcal{X}$ or $X:\Delta:\sigma:B\in\mathcal{X}$.*
5. *If $X:\Delta:\sigma:\Box A\in\mathcal{X}$, then the following conditions have to be satisfied, where it is determined by Table 3 which conditions apply for the logic* **L***:*

   (K) condition: *$X\cup\{n\}:\Delta:\sigma.(n):A\in\mathcal{X}$ for every $n\in\mathbf{N}$;*

   (4) condition: *$X\cup\{n\}:\Delta:\sigma.(n):\Box A\in\mathcal{X}$ for every $n\in\mathbf{N}$;*

   (4^d) condition: *if $\sigma=\tau.l$ then $X\cup\{n\}:\Delta:\sigma.(n):\Box A\in\mathcal{X}$ for all $n\in\mathbf{N}$;*

   (4^r) condition: *if $\sigma=\tau.l$ then $X:\Delta\cup\{\sigma\}:\tau:\Box A\in\mathcal{X}$;*

   (T) condition: *$X:\Delta:\sigma:A\in\mathcal{X}$;*

   (B) condition: *if $\sigma=\tau.l$ then $X:\Delta\cup\{\sigma\}:\tau:A\in\mathcal{X}$;*

   (5) condition: *if $\sigma=1.l$ then $X:\Delta\cup\{\sigma\}:1:\Box\Box A\in\mathcal{X}$.*

6. If $X:\Delta:\sigma:\Diamond A \in \mathcal{X}$ *then* $X:\Delta:\sigma.n:A \in \mathcal{X}$ *for some* $n \in \mathbf{N}$.

Lemma 37 *If* $\mathcal{X}$ *is an* $\mathbf{L}$*-Hintikka set, then there is an* $\mathbf{L}$*-interpretation* $\langle \mathbf{M}, \mathbf{I} \rangle$ $\mathbf{L}$*-satisfying* $\mathcal{X}$.

Proof. We define the $\mathbf{L}$-model $\mathbf{M} = \langle W, R, V \rangle$ as follows:

1. Put $W = \{[\sigma] \mid \sigma \in \mathcal{L}\}$ if $\mathbf{L}$ is serial and put

$$W = \{[\sigma] \mid \sigma \in \mathrm{lab}(\mathcal{X}), \sigma \text{ is justified in } \mathcal{X}\}$$

   if $\mathbf{L}$ is not serial, where $[\sigma]$ is the equivalence class of all labels that are identical to σ up to (conditional) parentheses.

2. For all $[\sigma], [\tau] \in W$, let $[\sigma]R[\tau]$ iff $\sigma \rhd \tau$; that is, iff τ is $\mathbf{L}$-accessible from σ (see Table 2).

3. For each primitive proposition p let $V(p)$ be defined by:

   - If $\mathbf{L}$ is serial, then $V(p) = \{[\sigma] \mid X:\Delta:\sigma:p \in \mathcal{X}\}$.
   - Otherwise, if $\mathbf{L}$ is not serial, then

$$V(p) = \{[\sigma] \mid X:\Delta:\sigma:p \in \mathcal{X}, \mathbf{I}(\sigma) \in W, \mathbf{I}(\xi) \in W \text{ for all } \xi \in \Delta\} \ .$$

The (identity) interpretation $\mathbf{I}$ is defined by:

$$\mathbf{I}(\sigma) = \begin{cases} [\sigma] & \text{if } [\sigma] \in W \\ \bot & \text{otherwise} \end{cases}$$

Lemma 12 then implies that $\mathbf{M}$ is an $\mathbf{L}$-model. According to its construction $\langle \mathbf{M}, \mathbf{I} \rangle$ is, therefore, an $\mathbf{L}$-interpretation.

We show by induction on the degree of tableau formulae $\phi = X:\Delta:\sigma:A$, that if $\phi \in \mathcal{X}$ then (a) $[\sigma] \notin W$, or (b) $[\xi] \notin W$ for some $\xi \in \Delta$, or $[\sigma] \models A$. This induction hypothesis implies that $\langle \mathbf{M}, \mathbf{I} \rangle$ $\mathbf{L}$-satisfies ϕ. When the induction proof is completed, we have shown that all formulae in $\mathcal{X}$ and thus $\mathcal{X}$ itself are $\mathbf{L}$-satisfied by the $\mathbf{L}$-interpretation $\langle \mathbf{M}, \mathbf{I} \rangle$.

Let the degree of a modal formula A be defined syntactically (as usual). The degree *deg* of a tableau formula is then defined by

$$deg(X:\Delta:\sigma:A) < deg(X':\Delta':\sigma':A') \text{ iff } \begin{cases} \text{(a) } deg(A) < deg(A') \text{ or} \\ \text{(b) } deg(A) = deg(A'), |\sigma| < |\sigma'| \end{cases}$$

Base case: The induction base are formulae ϕ where A is a literal. In case $A = p$, $[\sigma] \in W$, and $[\xi] \in W$ for all $\xi \in \Delta$, we have $[\sigma] \models p$ by the definition of V.

For the case $A = \neg p$, we have to show that if $[\sigma] \in W$ and $[\xi] \in W$ for all $\xi \in \Delta$ then $[\sigma] \models \neg p$. That is, $[\sigma] \notin V(p)$. For a contradiction suppose that $[\sigma] \in V(p)$. Then by the definition of V there has to be a tableau formula $X':\Delta':\sigma':p \in \mathcal{X}$ and, in addition, if $\mathbf{L}$ is not serial then $[\sigma'] \in W$ and $[\xi'] \in W$ for all $\xi' \in \Delta'$. By definition, if $\mathbf{L}$ is not serial, $[\sigma'] \in W$ iff σ' is justified in $\mathcal{X}$.

Thus we have two complementary and "completely justified" atomic formulae in $\mathcal{X}$; contradicting condition 2 in the definition of Hintikka sets.

The induction step is separated into cases according to the form of the formula A in ϕ:

$A = B \wedge C$: According to condition 3 in the definition of Hintikka sets, there are formulae $X : \Delta : \sigma : B \in \mathcal{X}$ and $X : \Delta : \sigma : C \in \mathcal{X}$. The induction hypothesis applies to these formulae. Therefore, (a) $[\sigma] \notin W$, or (b) $[\xi] \notin W$ for some $\xi \in \Delta$, or $[\sigma] \models B$ and $[\sigma] \models C$ which implies $[\sigma] \models B \wedge C$. This concludes the proof for this subcase.

$A = B \vee C$: Similar to $A = B \wedge C$.

$A = \Box B$: Suppose (a) $[\sigma] \in W$, and (b) $[\xi] \in W$ for all $\xi \in \Delta$; we then have to prove that $[\tau] \models B$ for all $[\tau] \in W$ such that $\sigma \rhd \tau$. We first show that (certain combinations of) the sub-conditions laid out as part of condition 5 of the definition of Hintikka sets imply this property for certain $[\tau] \in W$:

(K) condition: for all τ of the form $\tau = \sigma.[n]$ where $n \in \mathbf{N}$. Proof: There has to be a formula $X' : \Delta : \sigma.(n) : B \in \mathcal{X}$ for every $n \in \mathbf{N}$, that the induction hypothesis applies to. Thus, if $[\tau] \in W$, then $[\tau] \models B$.

(K) and (4) conditions: for all τ of the form $\tau_k = \sigma.n_1 \ldots n_k$ where $k \geq 1$ and $n_1, \ldots, n_k \in \mathbf{N}$. Proof: This requires an induction on k. We show, that for all $k \geq 0$ (we put $\tau_0 = \sigma$) there is a formula $X^k : \Delta : \tau_k : \Box B \in \mathcal{X}$ (for some X^k), which then, using the same argument as above for the (K) condition implies $[\tau_k] \models B$ if $[\tau_k] \in W$ for all $k \geq 1$. For $k = 0$ we have $X^0 : \Delta : \tau_0 : \Box B = X : \Delta : \sigma : A \in \mathcal{X}$ by assumption. Induction step: if $X^k : \Delta : \tau_k : \Box B \in \mathcal{X}$, then $X^{k+1} : \Delta : \tau_{k+1} : \Box B \in \mathcal{X}$ (with $X^{k+1} = X^k \cup \{n_k\}$), because the (4) condition applies.

(K) and (4^d) conditions: for all τ of the form $\tau_k = \sigma.n_1 \ldots n_k$ where $k \geq 1$ and $n_1, \ldots, n_k \in \mathbf{N}$, provided that $|\sigma| \geq 2$. Proof: similar to that for the (K) and (4) conditions.

(T) condition: for $\tau = \sigma$. Proof: There has to be a formula $X : \Delta : \sigma : B \in \mathcal{X}$, that the induction hypothesis applies to. Thus, if $[\sigma] \in W$, then $[\sigma] \models B$.

(B) condition: for τ such that $\sigma = \tau.l$. Proof: There has to be a formula $X : \Delta \cup \{\sigma\} : \tau : B$ in $\mathcal{X}$ that the induction hypothesis applies to. Thus, if $[\tau] \in W$, then $[\tau] \models B$.

(K), (4^d), (4^r), (5) conditions: if $|\sigma| \geq 2$ then for all τ such that $|\tau| \geq 2$, else (if $\sigma = 1$) then for all τ of the form $1.n$ where $n \in \mathbf{N}$. Proof: If $|\sigma| \geq 2$, then we prove by induction on the length of τ using the (4^r) condition that for all $\tau \in \mathrm{ipr}(\sigma)$ there is a formula $X : \Delta : \tau : \Box B \in \mathcal{X}$. Using the same argument as above for the (K) condition this implies $[\tau] \models B$ for all τ such that $|\tau| = 2$ and $[\tau] \in W$. In addition, we have $X : \Delta : 1.m : \Box B \in \mathcal{X}$ for some $m \in \mathbf{N}$ (where $1.m \in \mathrm{ipr}(\sigma)$; thus the (5) condition implies that $X : \Delta : 1 : \Box\Box B \in \mathcal{X}$ and (using the (K) condition) $X : \Delta : 1.n : \Box B \in \mathcal{X}$ for all $n \in \mathbf{N}$. Now, we can

procceed to prove $[\tau] \models B$ for all τ such that $|\tau| \geq 3$ and $[\tau] \in W$ as in the case of the (K) and (4) conditions. If $\sigma = 1$, then we can derive $[\sigma.n] \models B$ for all $n \in \mathbf{N}$ and $[\sigma.n] \in W$ using the (K) condition (see above).

(K), (4), (4^r) conditions: for all τ such that $|\tau| \geq 2$. Proof: We prove by induction on the length of τ using the (4^r) condition that for all $\tau \in \mathrm{ipr}(\sigma)$ there is a formula $X : \Delta : \tau : \Box B \in \mathcal{X}$. In particular, we have $X : \Delta : 1 : \Box B \in \mathcal{X}$. Now, we can procceed to prove $[\tau] \models B$ for all τ such that $|\tau| \geq 2$ and $[\tau] \in W$ as in the case of the (K) and (4) conditions.

(K), (T), (4), (4^r) conditions: for all τ. Proof: Similar to the case of conditions (K), (4) and (4^r) we prove by induction that for all τ such that $[\tau] \in W$ there is a formula $X : \Delta : \tau : \Box B \in \mathcal{X}$, which then, using the same argument as above for the (T) condition implies $[\tau] \models B$.

By checking Table 2, it becomes obvious that the sub-conditions for box-formulae that apply to an **L**-Hintikka set, indeed imply: if $[\tau] \in W$ and $\sigma \rhd \tau$, then $[\tau] \models B$.

$A = \Diamond B$**:** According to condition 6 in the definition of Hintikka sets, there is a formula $X : \Delta : \sigma.n : B \in \mathcal{X}$. The induction hypothesis applies to this formula. Therefore, (a) $[\sigma.n] \notin W$ or (b) $[\xi] \notin W$ for some $\xi \in \Delta$; or $[\sigma.n] \models B$. Now, since the label $\sigma.n$ itself occurs in $\mathrm{lab}(\mathcal{X})$, $[\sigma.n] \notin W$ implies that $\sigma.n$ is not justified in $\mathcal{X}$. Since the last position of $\sigma.n$ is unconditional this implies that σ is not justified in $\mathcal{X}$. Hence $[\sigma] \notin W$. Furthermore, if $[\sigma.n] \models B$ then $[\sigma] \models \Diamond B$ since $\sigma \rhd \sigma.n$ for all **L**. Together, we have enough to prove that (a) $[\sigma] \notin W$ or (b) $[\xi] \notin W$ for some $\xi \in \Delta$ or $[\sigma] \models \Diamond B$ as desired.

∎

6.2 Step 2: Definition of $\mathcal{T}_\infty$

Given a fair tableau procedure Ψ and an intial tableau $\mathcal{T}^0 = \emptyset : \emptyset : 1 : A$, let $\mathcal{T}_1$, $\mathcal{T}_2$, etc. be the tableau constructed using Ψ without closing branches or applying a substitution. These tableaux approximate the infinite tree $\mathcal{T}_\infty$.

6.3 Step 3: If $\mathcal{T}_\infty$ cannot be closed then ...

Now suppose that there is no substitution that allows $\mathcal{T}_\infty$ to be closed. Thus we can choose any substitution θ we like, and we are guaranteed that $\mathcal{T}_\infty\theta$ will contain some open branch. The branch may differ according to the choice of θ.

We now define a particular substitution θ_∞ as follows: Let $\{\mathcal{B}_1, \mathcal{B}_2, \ldots\}$ be an enumeration of the branches of $\mathcal{T}_\infty$. Let $\Phi = \{\phi_1, \phi_2, \ldots\}$ be an enumeration of the disjunctive formulae (formulae of the form $X_i : \Delta_i : \sigma_i : B_i \vee C_i$) in $\mathcal{T}$ *excluding renamings*. For every disjunctive formula, if ϕ_i occurs on $\mathcal{B}_k$ then let ϕ_{ijk} be the j-th renaming of ϕ_i on the k-th branch.

Now, the label σ_i of ϕ_i will be of finite length. Therefore, the set of all ground instances of τ_i is enumerable; let $\{\sigma_i^1, \sigma_i^2, \ldots\}$ be such an enumeration.

Let x be a variable in X_{ijk} and suppose it occurs in the p-th position of σ_{ijk}. Note, that the sets X_{ijk} of universal variables are all pairwise disjoint even if only one of the formulae in the numerator of the conjunctive rule is renamed. To ensure that the k-th branch $\mathcal{B}_k$ is a potential source of an **L**-model we have to ensure that the occurrences of ϕ_i on $\mathcal{B}_k$ "cover" all the instances of σ_i. To this end choose θ_∞ so that: $\theta_\infty(x) = n$, where n is the value of the p-th position of σ_i^j. Thus if x is in the p-th position of σ_{i1k} its value "covers" σ_i^1, if it is in the p-th position of σ_{i2k} its value "covers" σ_i^2, and so on.

Lemma 38 *If θ_∞ is defined as above, and $\mathcal{B}$ is an open branch of the tableau $\mathcal{T}_\infty$ that cannot be closed when θ_∞ is applied then*

$$\mathcal{X} = \{\phi\lambda_{|X}\theta_\infty \mid \phi = X:\Delta:\sigma:A \text{ is a formula on } \mathcal{B}, \text{ and } \lambda \text{ is a grounding substitution}\}$$

is an **L***-Hintikka set.*

Proof. We have to check each clause of Definition 36 for the set $\mathcal{X}$.

1. It is obvious that the root is 1, and fairly easy to see that we always produce a strongly generated set.

2. Suppose condition 2 in the definition of Hintikka sets is violated by $\mathcal{X}$. Then there have to be formulae $\phi_1 = X_1:\Delta_1:\sigma_1:p$, $\phi_2 = X_2:\Delta_2:\sigma_2:\neg p$ on $\mathcal{B}$ and grounding substitutions λ_1 and λ_2, such that $\sigma_1\lambda_{1|X_1}\theta_\infty = \sigma_2\lambda_{2|X_2}\theta_\infty = \varsigma$, and (b) the logic **L** is serial or all labels in $\{\varsigma\}\cup\Delta_1\lambda_{1|X_1}\theta_\infty\cup\Delta_2\lambda_{2|X_2}\theta_\infty$ are justified in $\mathcal{X}$. Our expansion rules always use $\mathcal{T}$-renamings of universal variables in their numerator, hence $X_1 \cap X_2 = \emptyset$. Therefore, there is a single grounding substitution λ of the universal variables in $\mathcal{T}_\infty$, such that $\sigma_1\lambda = \sigma_1\lambda_{1|X_1}$ and $\sigma_2\lambda = \sigma_2\lambda_{2|X_2}$. In addition, $\lambda \circ \theta_\infty = \theta_\infty \circ \lambda$, since θ_∞ only instantiated free variables in $\mathcal{T}_\infty$. Thus the branch $\mathcal{B}$ *can* be closed using the substitution λ of the universal variables in $\mathcal{T}_\infty$, which contradicts the choice of $\mathcal{B}$.

3. We have to show, that for all $\phi = X:\Delta:\sigma:A \wedge B \in \mathcal{B}$ and all grounding substitutions λ, and thus for all formulae of the form $\phi\lambda_{|X}\theta_\infty$, the formulae $\phi_1\lambda_{|X}\theta_\infty$ and $\phi_2\lambda_{|X}\theta_\infty$ are in $\mathcal{X}$, where $\phi_1 = \emptyset:\Delta:\sigma:A$ and $\phi_2 = \emptyset:\Delta:\sigma:B$. Since the appropriate rule has been applied to ϕ, the formulae $X:\Delta:\sigma:A$ and $X':\Delta':\sigma':B$ are both on $\mathcal{B}$. Thus, $\phi_1\lambda_{|X}\theta_\infty \in \mathcal{X}$. To prove the same for ϕ_2, let μ be the variable renaming such that $X':\Delta':\sigma':B = (X:\Delta:\sigma:B)\mu$, and put $\lambda' = (\lambda\circ\mu)$, which implies $\lambda'_{|X'} = \lambda_{|X}\circ\mu$. The substitution λ' is grounding. Therefore, by definition the set $\mathcal{X}$ contains the formula $\phi_2'\lambda'_{|X'}\theta_\infty$, which is identical to $\phi_2\lambda_{|X}\theta_\infty$.

4. We have to show, that for all $\phi = X:\Delta:\sigma:A \vee B \in \mathcal{B}$ and all grounding substitutions λ, and thus for all formulae of the form $\phi\lambda_{|X}\theta_\infty$, one of the formulae $\phi_1\lambda_{|X}\theta_\infty$ and $\phi_2\lambda_{|X}\theta_\infty$ is in $\mathcal{X}$, where $\phi_1 = \emptyset:\Delta:\sigma:A$ and $\phi_2 =$

$\emptyset:\Delta:\sigma:B$. If $X = \emptyset$ then this holds immediately by the special case of the disjunctive rule. Otherwise, according to the construction of $\mathcal{T}_\infty$ and θ_∞, there has to be a renaming $\phi' = \phi\rho$ of ϕ on $\mathcal{B}$ (where ρ is the renaming substitution) such that $\theta_{\infty|X'} \circ \rho = \lambda_{|X}$, i.e., $\phi'\theta_\infty = \phi\lambda_{|X}\theta_\infty$. Since the appropriate rule has been applied to ϕ', one of the formulae $\phi'_1 = \phi_1\rho$ and $\phi'_2 = \phi_2\rho$ is on $\mathcal{B}$ and thus $\phi'_1\lambda_{|\emptyset}\theta_\infty = \phi'_1\theta_\infty = \phi_1\lambda_{|X}\theta_\infty$ or $\phi'_2\lambda_{|\emptyset}\theta_\infty = \phi_2\lambda_{|X}\theta_\infty$ is in $\mathcal{X}$.

Since the other conditions in the definition of Hintikka sets closely resemble the tableau expansion rules, the proof that these conditions hold for $\mathcal{X}$ is similar to that for conjunctive formulae. ∎

Lemma 39 *If there is no substitution that closes the tableau $\mathcal{T}_\infty$ for $\emptyset:\emptyset:1:A$, then A is* **L***-satisfiable.*

Proof. Since $\mathcal{T}_\infty$ cannot be closed for any substitution, it cannot be closed for θ_∞ as defined above. Thus there is some open branch in $\mathcal{T}_\infty\theta_\infty$. By Lemma 38 this branch forms an **L**-Hintikka set. By Lemma 37 such a set gives an **L**-interpretation $\langle \mathbf{M}, \mathbf{I}\rangle$ that **L**-satisfies the root $\emptyset:\emptyset:1:A$ of $\mathcal{T}^0$. But this means that in the **L**-model **M** we must have $\mathbf{I}(1) \models A$. ∎

6.4 Step 4

The contrapositive of this lemma is: If A is **L**-unsatisfiable, then there is at least one substitution θ that, when applied, allows to close all branches in $\mathcal{T}_\infty$. Thus there is an n such that all branches in the finite tableau $\mathcal{T}_n\theta$ can be closed. We construct another tableau $\mathcal{T}'_n$ from $\mathcal{T}_n$ by starting at the end of each branch and removing all formulae that are not needed to close the corresponding branch in $\mathcal{T}_n\theta$ (that includes justification). Note that $\mathcal{T}'_n\theta$ is still closed but is unlikely to be of uniform depth.

6.5 Step 5:

Lemma 40 *If $\mathcal{T}'_n\theta$ is closed for some finite n, then the substitution θ can be decomposed so that: $\theta = \theta' \circ \xi_r \circ \xi_{r-1} \circ ... \circ \xi_1$ where ξ_i is a most general closing substitution for the instantiation $(\mathcal{B}_i)\xi_1\xi_2...\xi_{i-1}$ of the i-th branch, $\mathcal{B}_i$, in $\mathcal{T}'_n$. And θ' is the part of θ that is not actually needed to close $\mathcal{T}'_n$.*

Proof. We construct the ξ_i inductively as follows:

Define $\xi'_0 = \theta$. For $1 \le i \le r$, let ξ_i be a most general substitution, such that

1. ξ'_{i-1} is a specialisation of ξ_i; that is, there is a substitution ξ'_i such that $\xi'_{i-1} = \xi'_i \circ \xi_i$; and
2. ξ_i is a closing substitution for $(\mathcal{B}_i)\xi_1\xi_2...\xi_{i-1}$.

Then ξ_i is a most general *closing* substitution. For otherwise, there must be a closing substitution ξ''_i, that is more general than ξ_i. The is-more-general relation is transitive hence ξ''_i is more general than ξ'_{i-1}, which contradicts our choice of ξ_i as a most general substitution satisfying the two conditions.

Finally, define $\theta' = \xi'_r$. ∎

6.6 Step 6

Since Ψ is a fair tableau procedure we have proved the completeness theorem as stated in the paper.

Our implementation uses a fair tableau procedure and uses backtracking to resolve any remaining non-determinism; namely which closing substitution to choose, and whether to close a branch (if possible) or to expand it. Thus it will find the proof in a finite amount of time.

7 leanK: An Implementation

We have implemented our calculus as a "lean" theorem prover written in Prolog (the source code is available at `http://i12www.ira.uka.de/modlean` on the *World Wide Web*). The basic version for the logic **K** is called leanK, and consists of just eleven Prolog clauses and 45 lines of code. The version for the logic **KD** which does not demand justified labels, is even shorter: it consists of only 6 clauses and 27 lines of code. (Below we describe this version for the logic **KD** only.)

We use Prolog syntax for formulae: primitive propositions are Prolog terms, "`-`" is negation, "`;`" disjunction, "`,`" conjunction, the prefix-operator "`box`" is the box-operator, and "`dia`" is the diamond-operator. Thus, a modal formula is represented by a Prolog term (for example, the formula $p \wedge \Box(\neg p \vee \Diamond p)$ is represented by `(p,box(-p;dia p))`).

The Prolog predicate

```
prove(Fml,Label,Univ,Lits,UnExp,Free,Limit)
```

implements our prover; it succeeds if there is a closed tableau (of a certain size) for the formula bound to `Fml`. The prover is started with the goal

```
prove(Fml,[],[],[],[],[],Limit)
```

which succeeds if `Fml` can be proven inconsistent without using more than `Limit` free variables on each tableau branch.[3]

The meaning of the arguments of `prove` is:

`Fml`: The current formula.

`Label`: The label of the current formula.

`Univ`: The list of universal variables in the current formula.

[3] If one wants to avoid committing on the number `Limit`, the predicate `prove` can be called with iterative deepening on `Limit`. The standard solution in Prolog for this is:

```
inc_prove(Fml,Limit) :- prove(Fml,[],[],[],[],[],Limit).
inc_prove(Fml,Limit) :- NewLimit is Limit + 1,
                        inc_prove(Fml,NewLimit).
```

When started with `inc_prove(Fml,N)`, the prover searches with the values `N`, `N+1`, ... for `Limit`.

`Lits`: The set of literals on the branch; in case the current formula is a literal ϕ, this list contains only those literals, that have not been used yet to be unified with ϕ to close the branch. A literal is stored in the form `(Label:Neg)`, where `Neg` is the complement of the literal, and `Label` is its label.

`UnExp`: The set of formulae that have not been considered yet. Formulae are stored in the form `(Univ:Label:Fml)`.

`Free`: A Prolog term containing all free variables introduced in labels (these can be instantiated).

`Limit`: The number of variables, that may still be added (has to be greater than 0).

The conjunctive rule:

```
prove((A,B),Label,Univ,Lits,UnExp,Free,Limit) :- !,
        copy_term((Label,Univ,Free),(LabelB,UnivB,Free)),
        prove(A,Label,Univ,Lits,[(UnivB:LabelB:B)|UnExp],Limit).
```

The disjunctive rule:

```
prove((A;B),Label,Univ,Lits,UnExp,Free,Limit) :- !,
        Limit >= 0,
        copy_term((Label,Univ,Free),(LabelA,UnivA,Free)),
        copy_term((Label,Univ,Free),(LabelB,UnivB,Free)),
        append(UnExp,[UnivA:LabelA:(A;B)],UnExpA),
        append(UnExp,[UnivB:LabelB:(A;B)],UnExpB),
        length(Univ,Length),
        NewLimit is Limit - Length,
        prove(A,Label,[],Lits,UnExpA,(Univ+Free),NewLimit),
        prove(B,Label,[],Lits,UnExpB,(Univ+Free),NewLimit).
```

The box-rule:

```
prove(box Fml,Label,Univ,Lits,UnExp,Free,Limit) :- !,
        prove(Fml,[X|Label],[X|Univ],Lits,UnExp,Free,Limit).
```

The diamond-rule:

```
prove(dia Fml,Label,Univ,Lits,UnExp,Free,Limit) :- !,
        prove(Fml,[Fml|Label],Univ,Lits,UnExp,Free,Limit).
```

This clause applies if the current formula is a literal, and tries to close the branch:

```
prove(Lit,Label,_,[L|Lits],_,Free,_) :-
        copy_term(((Label:Lit),Free),(New,Free)),
        (    copy_term((L,Free),(New,Free)),
          retract(branches(B)), NB is B+1, assert(branches(NB))
        ;    prove(Lit,Label,_,Lits,[],[],_,_)
        ).
```

This clause applies, if there is no literal left on the branch, that one could try to unify the current formula with:

```
prove(Lit,LitLabel,_,Lits,[(Univ:Label:Fml)|UnExpR],Free,Limit) :-
        !,
        ( (Lit = -Neg; -Lit = Neg) ->
          prove(Fml,Label,Univ,[(LitLabel:Neg)|Lits],
                UnExpR,Free,Limit)
        ).
```

The leanK program employs the following fair tableau procedure: Given a tableau $\mathcal{T}$, the branch that is expanded next is the left-most open branch, with the formulae on any particular branch organised as a queue. The first formula in the chosen branch/queue is removed from the queue and is used to update the tableau as follows:

- If the chosen formula is not a literal then some (one) rule is applicable to it, and the formulae created by that rule application are added to the queue as follows: if the (traditional part of the) created formula is strictly less complex than the premiss, this new formula is added to the front of the queue, otherwise it is added to the end of the queue. In particular, this means that renamings of formulae added by the disjunctive rule, and the formula labelled (4) and (4^r) in the numerator of the box-rule, are added to the end of the queue.

- If the chosen formula in the queue is a literal ϕ and there is a most general substitution μ of the free variables in ϕ such that $\phi\mu$ and some other literal $\psi\mu$ on the branch can be used for closure, then there is a choice point: (1) the substitution μ may be applied and the branch closed, or (2) the literal is removed from the queue and the next formula moves to the front. There is a further choicepoint if there is more than one closing substitution μ. In case no closing substitution μ exists, option (2) is used deterministically. If there is a choice, Prolog's backtracking mechanism is used to resolve this non-determinism and explore all choices.

Since this procedure is essentially a depth first search, a limit is imposed on the number of free variables in a branch, thereby forcing every branch to terminate after some finite number of rule applications. Prolog's backtracking mechanism then automatically processes the next branch in the queue. Iterative deepening is used to preserve completeness by increasing this branch limit, step by step, as long as no proof can be found.

A lean and efficient implementation is only possible by making use of Prolog's special features: Prolog's backtracking is used to resolve the non-determinism in the tableau procedure; Prolog's built-in unification is used for finding most general closing substitutions and for the justification test; and Prolog's indexing mechanism is employed to quickly determine the appropriate tableau rule for the next formula.

To avoid generating useless renamings of disjunctive formulae, the version of leanK used to obtain Table 4 uses the following restriction: when the disjunctive rule is applied to a formula $\phi = X : \Delta : \sigma : A \vee B$, the (potentially useless) renaming ϕ' created by the disjunctive rule is "protected" from further applications of the disjuntive rule until one of the variables in X has been instantiated. That is,

No.	24	44	46	50	52	55	56	67	72
Branches	22251	90	137	43	56	1011	68	26565	154
Var.-Limit	10	5	5	4	4	11	4	11	6
Time [msec]	4400	50	80	20	30	1000	30	9520	90

Table 4: Statistics for set of **K**-theorems.

a renaming is useful only when one of the original variables in X has been used to close a branch using a descendant of $\emptyset : \Delta : \sigma : A$ or $\emptyset : \Delta : \sigma : B$.

Table 4 shows statistics for a set of 72 **K**-theorems kindly provided by Alain Heuerding. Of these, leanK could prove 61 in the allotted time of 15 seconds, with 52 in less than 10msec (not shown in the table). The program was terminated if no proof had been found after 15 seconds. The table shows the number of branches that were closed, the maximal number of free variables in a branch, and the proof time (running under SICStus Prolog on a SUN Ultra 1 workstation).

The examples that took several seconds to prove show an advantage of lean implementations: the very high inference rate of about 2500 closed branches per second. The complexity of these formulae is non-trivial; one of the more complex ones, No. 55, is:

$$((\Box(\Box(p\rightarrow\Box p)\rightarrow p)\rightarrow p)\wedge(\Box(\Box((\Box(p\rightarrow\Box p)\rightarrow p)\wedge(\Box(\Box(p\rightarrow\Box p)\rightarrow p)\rightarrow$$
$$\Box\Box(\Box(p\rightarrow\Box p)\rightarrow p))\rightarrow\Box((\Box(p\rightarrow\Box p)\rightarrow p)\wedge(\Box(\Box(p\rightarrow\Box p)\rightarrow p)\rightarrow\Box\Box(\Box(p\rightarrow\Box p)\rightarrow p))))\rightarrow$$
$$(\Box(p\rightarrow\Box p)\rightarrow p)\wedge(\Box(\Box(p\rightarrow\Box p)\rightarrow p)\rightarrow\Box\Box(\Box(p\rightarrow\Box p)\rightarrow p)))\rightarrow(\Box(p\rightarrow\Box p)\rightarrow p)\wedge$$
$$(\Box(\Box(p\rightarrow\Box p)\rightarrow p)\rightarrow\Box\Box(\Box(p\rightarrow\Box p)\rightarrow p)))\rightarrow\Box(\Box(p\rightarrow\Box p)\rightarrow p)\rightarrow p\vee\Box p)$$

Note that although the above formula contains many occurrences of the same sub-formula, it is not just a complex instance of a "vanilla" **K**-theorem. The fact that leanK closes branches only on literals rather than on arbitrary complex formulae means that these complex sub-formula occurrences hinder rather than help leanK.

8 Conclusion and Future Work

Our initial results, presented in the last section, are very encouraging. We believe that labels with variables deliver the following advantages:

- The use of variables generates a smaller search space since a label can now stand in for all its ground instances. This is in stark contrast to the modular systems of [16, 10], where only ground labels are used.

- The use of a Gödelisation function in the diamond-rule leads to a smaller number of labels than in other labelled tableau methods since two different occurrences of the formula $X : \Delta : \sigma : \Diamond A$ lead to the same formula $X : \Delta : \sigma.\lceil A\rceil : A$. We therefore do not need to delete duplicate occurrences of a formula as is done in some tableau implementations for modal logics. This is particularly important since the world $\sigma.\lceil A\rceil$ may be the root of a large sub-model and duplicating it is likely to be extremely inefficient.

– Our "lean" implementation is perfect for applications where the deductive engine must be transparent and easily modifiable.

Our method is really a very clever translation of propositional modal logics into first-order logic, and most of the complications arise because some worlds may have no successors. The new notion of conditional labels allows us to keep track of these complications, and thus handle the non-serial logics that frustrate other "general frameworks" [8, 13]. Nevertheless, our method can also handle *second order* "provability" logics like **G** and **Grz**; see [10]. Furthermore, specialised versions of these tableau systems can match the theoretical lower bounds for particular logics like **K45**, **G** and **Grz** if we give up modularity; see [10, 16]. We intend to extend our initial implementation of leanK along these lines.

The 15 basic modal logics are known to be decidable and techniques from [6, 10, 16, 12] can be used to extend our method into a decision procedure. However, it is not clear that this is possible in a *lean* way. The extra implementation restriction mentioned in the previous section is of vital importance here since it is essentially a demand driven *contraction rule* on box-formulae since box-formulae get copied only as the required free variables get instantiated. And controlling contraction is often the key to decidability.

Fitting [5] has recently shown how to view the original leanTAP program for classical propositional logic as an unusual sequent calculus `dirseq`. He has also shown how to extend `dirseq` to handle the modal logics **K**, **KT**, **K4**, and **S4**. As with traditional modal tableaux, however, `dirseq` does not handle the symmetric logics like **S5** and **B**. We are currently extending our work to give a modular free variable version of `dirseq` that does handle these logics.

It is also possible to extend our method to deal with the notions of global and local logical consequence [6].

An alternative variable label approach [11] uses different unification algorithms to find complementary literals for branch closure. However, the interactions between modalities, variable labels, and unification algorithms is by no means easy to disentangle. Extending our method to utilise special unification algorithms is perfectly possible, now that correctness and completeness have been worked out for the interactions between modalities and variable labels alone.

References

[1] Bernhard Beckert, Reiner Hähnle, Peter Oel, and Martin Sulzmann. The tableau-based theorem prover $_3$TAP, version 4.0. In *Proceedings, 13th International Conference on Automated Deduction (CADE), New Brunswick, NJ, USA*, LNCS 1104, pages 303–307. Springer, 1996.

[2] Bernhard Beckert and Joachim Posegga. leanTAP: Lean tableau-based deduction. *Journal of Automated Reasoning*, 15(3):339–358, 1995.

[3] E. Bencivenga. Free logic. In D. Gabbay and F. Günthner, editors, *Handbook of Philosophical Logic*, volume 3. Kluwer, Dordrecht, 1986.

[4] Marcello D'Agostino, Dov Gabbay, and Alessandra Russo. Grafting modalities onto substructural implication systems. *Studia Logica*, 1996. To appear.

[5] Melvin Fitting. Leantap revisited. Draft Manuscript, January 1996.

[6] Melvin C. Fitting. *Proof Methods for Modal and Intuitionistic Logics*, volume 169 of *Synthese Library*. D. Reidel, Dordrecht, Holland, 1983.

[7] Melvin C. Fitting. *First-Order Logic and Automated Theorem Proving*. Springer, second edition, 1996.

[8] Alan Frisch and Richard Scherl. A general framework for modal deduction. In J. Allen, R. Fikes, and E. Sandewall, editors, *Proc., 2nd Conference on Principles of Knowledge Representation and Reasoning*. Morgan-Kaufmann, 1991.

[9] Dov Gabbay. *Labelled Deductive Systems*. Oxford University Press, 1996. To appear.

[10] Rajeev Goré. Tableau methods for modal and temporal logics. In Marcello D'Agostino, Dov Gabbay, Reiner Hähnle, and Joachim Posegga, editors, *Handbook of Tableau Methods*, chapter 7. Kluwer, Dordrecht, 1997. To appear.

[11] Guido Governatori. A reduplication and loop checking free proof system for S4. In *Short Papers: TABLEAUX'96*, number 154-96 in RI-DSI, Via Comelico 39, 20135 Milan, Italy, 1996. Department of Computer Science, University of Milan.

[12] Alain Heuerding, Michael Seyfried, and Heinrich Zimmermann. Efficient loop-check for backward proof search in some non-classical logics. In P. Miglioli, U. Moscato, D. Mundici, and M. Ornaghi, editors, *Proceedings, 5th Workshop on Theorem Proving with Analytic Tableaux and Related Methods, Terrasini, Palermo, Italy*, LNCS 1071, pages 210–225. Springer, 1996.

[13] Peter Jackson and Hans Reichgelt. A general proof method for first-order modal logic. In *9th Int. Joint Conference on Artificial Intelligence*, pages 942–944, 1987.

[14] S. Kanger. *Provability in Logic*. Stockholm Studies in Philosophy, University of Stockholm. Almqvist and Wiksell, Sweden, 1957.

[15] Reinhold Letz, Johann Schumann, Stephan Bayerl, and Wolfgang Bibel. SETHEO: A high-performance theorem prover. *Journal of Automated Reasoning*, 8(2):183–212, 1992.

[16] Fabio Massacci. Strongly analytic tableaux for normal modal logics. In A. Bundy, editor, *Proceedings, 12th International Conference on Automated Deduction (CADE), Nancy, France*, LNCS 814, pages 723–737. Springer, 1994.

[17] Grigori Mints. *A Short Introduction to Modal Logic*. CSLI, Stanford, 1992.

[18] Steve V. Reeves. Semantic tableaux as a framework for automated theorem-proving. In C. Mellish and J. Hallam, editors, *Advances in Artificial Intelligence (Proceedings of AISB-87)*, pages 125–139. Wiley, 1987.

[19] Alessandra Russo. Generalising propositional modal logic using labelled deductive systems. In F. Baader and K. Schulz, editors, *Proceedings, Frontiers of Combining Systems (FroCoS), Munich, Germany*, volume 3 of *Applied Logic Series*. Kluwer, Dordrecht, 1996.