A Uniform Proof Procedure for Classical and Non-Classical Logics

Jens Otten* Christoph Kreitz

Fachgebiet Intellektik, Fachbereich Informatik
Technische Hochschule Darmstadt
Alexanderstr. 10, 64283 Darmstadt, Germany
{jeotten,kreitz}@intellektik.informatik.th-darmstadt.de

Abstract. We present a proof procedure for classical and non-classical logics. The proof search is based on the matrix-characterization of validity where an emphasis on paths and connections avoids redundancies occurring in sequent or tableaux calculi. Our uniform path-checking algorithm operates on arbitrary (non-normal form) formulae and generalizes Bibel's connection method for classical logic and formulae in clause-form. It can be applied to intuitionistic and modal logics by modifying the component for testing complementarity of connected atoms. Besides a short and elegant path-checking procedure we present a specialized string-unification algorithm which is necessary for dealing with non-classical logics.

1 Introduction

Non-classical logics are used extensively in various branches of AI and Computer Science as logics of knowledge and belief, logics of programs, and for the specification of distributed and concurrent systems. In many of these applications there is a need for automated proof search. For *classical* predicate logic theorem provers based on resolution [14, 17], the connection method [4, 5, 8, 3], or the tableaux calculus [2, 1] have demonstrated that formal reasoning can be automated well.

Recently Wallen [15, 16] and Ohlbach [10] have extended the classical characterizations of logical validity on which the above proof methods are based into characterizations of logical validity for intuitionistic and various modal logics. These characterizations avoid the notational redundancies contained in natural deduction or sequent calculi for these logics and thus provide a theoretical foundation for developing proof procedures for a rich variety of non-classical logics.

In this paper we present a proof procedure which allows a *uniform* treatment of classical, intuitionistic, and modal logics. It is based on a unified representation of Wallen's matrix characterizations and generalizes Bibel's connection method [4, 5] for classical predicate logic accordingly. In order to keep the general methodology of the latter – following *connections* when investigating *paths* through a given formula – unchanged we had to take into account a considerable extension of the notion of *complementarity* which strongly depends on the logic under consideration. Whereas in the classical case two atomic formulae are complementary if they have different polarity but can be unified by some term-substitution, for non-classical logics also the *prefixes* of the two atomic formulae (i.e. descriptions of their position in the formula tree) have to be unifiable. Thus in addition to the usual term-unification our method requires a special *string-unification* for making two prefixes identical. The absence of normal-forms for most non-classical logics also makes it necessary to deal with a nested matrix instead of a clause-form. For

* Supported by the Adolf Messer Stiftung

In G. Görz, S. Hölldobler, ed., *KI-96: Advances in Artificial Intelligence*, LNAI 1137, pp. 307–319, Springer Verlag, 1996.

this purpose we had to extend the notions of *actual clauses* and *active paths* which guide the search for a classical matrix-proof in normal-form.

The resulting proof procedure is based on ideas originally developed for a generalized connection based proof method for intuitionistic logic [11]. It consists of a connection driven algorithm which checks the complementarity of all paths through a given formula and uses a component for testing complementarity. The latter depends on the underlying logic and is based on the string-unification procedure developed in [12]. This modular design allows us to treat a rich variety of logics in a uniform and simple way.

Our paper is organized as follows. In section 2 we resume fundamental concepts and provide uniform matrix characterizations of logical validity for classical, intuitionistic and various modal logics. In section 3 we introduce the concepts which are used for guiding the proof search and present our uniform procedure. In section 4 we discuss explicit methods for testing complementarity in various logics, particularly the specialized string-unification algorithm. We conclude with a few remarks on possible extensions.

2 Matrix Characterizations of Logical Validity

After describing some basic notations we shall present a matrix-characterization of validity for classical logic pioneered by Bibel [5] and explain the modifications which are necessary for extending this characterization to intuitionistic and various modal logics. We refer to [16] for detailed explanations.

2.1 Preliminaries

We assume the reader to be familiar with the language of classical and modal first-order logic. To get a uniform notation we firstly define formula trees, polarities, types, and multiplicities.

Definition 1 (Formula Tree, Positions, Labels). A *formula tree* is the representation of a formula as a tree. Each node is marked with a unique name, its *position* (denoted by $a_0, a_1, \ldots$). By $\mathcal{P}os$ we denote the set of all positions in the formula tree. The mapping $\mathrm{lab}(u)$ assigns to each position $u \in \mathcal{P}os$ in the tree its *label*. A position labeled with an atom is called an *atomic position*. The *tree ordering* $< \subseteq \mathcal{P}os \times \mathcal{P}os$ is the (partial) ordering on the positions in the formula tree, i.e. $a_i < a_j$ if (and only if) the position a_i is shown below a_j in the formula tree.

Example 1. Figure 1 shows the formula tree for $F_1 \equiv \forall x Px \Rightarrow Pa \wedge Pb$. It is, e.g., $\mathrm{lab}(a_0) = \text{`}\Rightarrow\text{'}$, $\mathrm{lab}(a_4) = \text{`}Pa\text{'}$, $a_1 < a_2$ and $a_0 < a_4$.

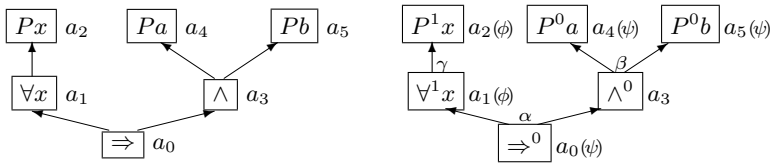

Fig. 1. Formula tree for F_1 (in the right tree nodes are marked with polarity/types)

Remark. Note that each subformula of a given formula corresponds to exactly one position in its formula tree, e.g. $\forall x Px$ corresponds to the position a_1.

Definition 2 (Polarity, Types).

1. The *polarity* of a position is determined by the label and polarity of its (direct) predecessor in the formula tree. The root-position has polarity 0.
2. The *principal type* of a position is determined by its label and polarity. Atomic positions have no principal type.
3. The *intuitionistic type* of a position is determined by its label and polarity. Only certain positions (atoms, $\Rightarrow$, $\neg$, $\forall$) have an intuitionistic type.

Polarities and types of positions are defined in table 1. For example, a position labeled with $\Rightarrow$ and polarity 1 has principal type β and intuitionistic type ϕ. Its successor positions have polarity 0 and 1, respectively.

principal type α	$(A \wedge B)^1$	$(A \vee B)^0$	$(A \Rightarrow B)^0$	$(\neg A)^1$	$(\neg A)^0$
sucessor polarity	A^1, B^1	A^0, B^0	A^1, B^0	A^0	A^1
principal type β	$(A \wedge B)^0$	$(A \vee B)^1$	$(A \Rightarrow B)^1$		
sucessor polarity	A^0, B^0	A^1, B^1	A^0, B^1		

principal type γ	$(\forall x A)^1$	$(\exists x A)^0$	*principal type* δ	$(\forall x A)^0$	$(\exists x A)^1$
sucessor polarity	A^1	A^0	sucessor polarity	A^0	A^1
principal type ν	$(\Box A)^1$	$(\Diamond A)^0$	*principal type* π	$(\Box A)^0$	$(\Diamond A)^1$
sucessor polarity	A^1	A^0	sucessor polarity	A^0	A^1

intuitionistic type ϕ	$(\neg A)^1$	$(A \Rightarrow B)^1$	$(\forall x A)^1$	P^1 (P is atomic)
intuitionistic type ψ	$(\neg A)^0$	$(A \Rightarrow B)^0$	$(\forall x A)^0$	P^0 (P is atomic)

Table 1. Polarity, principal type and intuitionistic type of positions

Example 2. The formula tree for F_1 where each position is marked with its polarity and principal type is given in figure 1. The positions a_1 and a_2 have intuitionistic type ϕ, positions a_0, a_4 and a_5 have intuitionistic type ψ.

Remark. For a given formula we denote the sets of positions of type α, β, γ, δ, ν, π, ϕ, and ψ by α, β , Γ, Δ, ν, Π, Φ, and Ψ.

A multiplicity encodes the number of distinct instances of subformulae to be considered during the proof search.

Definition 3 (Multiplicities).

1. The *first-order multiplicity* $\mu_Q : \Gamma \to \mathbb{N}$,
2. the *intuitionistic multiplicity* $\mu_J : \Phi \to \mathbb{N}$, and
3. the *modal multiplicity* $\mu_M : \nu \to \mathbb{N}$

assigns each position of type γ, ϕ and ν, respectively, a natural number.

Remark. By F^μ we denote an *indexed formula*, i.e. a formula and its multiplicity. We consider multiple instances of subformulae according to the multiplicity of its corresponding position in the formula tree and extend the notion of tree ordering accordingly. The new nodes in the formula tree get new positions but inherit the polarity and types from their source positions.

Example 3. The formula tree for F_1 with $\mu_Q(a_1)$=2 is shown in figure 2.[2]

Remark. For technical reasons we replace free variables in atomic formulae by their quantifier positions. Thus γ- and δ-positions appear in atomic formulae.

[2] Adopting Bibel's notion of multiplicity (instead of Wallen's) we copy the positions of type γ, ϕ or ν instead of their successors.

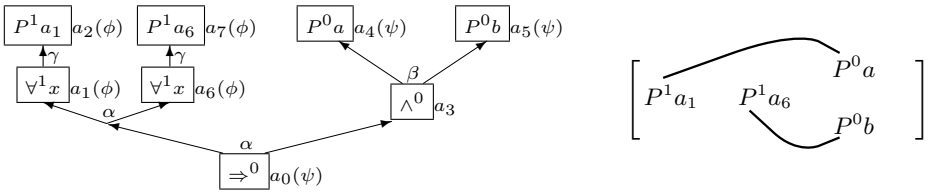

Fig. 2. Formula tree and matrix for F_1^μ and $\mu_Q(a_1)$=2

2.2 Classical Logic

To resume the characterization for classical logic [4, 5] we introduce the concepts of paths and connections. For the first-order (classical) logic it is further necessary to define the notion of a first-order substitution and the reduction ordering.

Definition 4 (Matrix,Path,Connection).

1. In the *matrix(-representation)* of a formula F we place the components of subformulae of principal type α side by side whereas components of subformulae of principal type β are placed one upon the other. Furthermore we omit all connectives and quantifiers.
2. A *path* through a formula F is a subset of the atomic positions of its formula tree; it is a horizontal path through the matrix representation of F.
3. A *connection* is a pair of atomic positions labeled with the same predicate symbol but with different polarities.

Example 4. The matrix for $F_1 \equiv \forall x Px \Rightarrow Pa \wedge Pb$ is given in figure 2. There are two paths through F_1, namely $\{a_2, a_7, a_4\}$ and $\{a_2, a_7, a_5\}$ ($\{Pa_1, Pa_6, Pa\}$ and $\{Pa_1, Pa_6, Pb\}$ in the notation of labels). The paths contain the connections $\{a_2, a_4\}$, $\{a_7, a_4\}$ and $\{a_2, a_5\}$, $\{a_7, a_5\}$, respectively.

Definition 5 (First-order Substitution, Reduction Ordering).

1. A *first-oder substitution* $\sigma_Q : \Gamma \to \mathcal{T}$ assigns to each position of type γ a term $t \,\epsilon\, \mathcal{T}$ (the set of all terms) where again variables in t are replaced by positions of type γ and δ.[3] A connection $\{u, v\}$ is said to be σ_Q*-complementary* iff $\sigma_Q(\mathrm{lab}(u)) = \sigma_Q(\mathrm{lab}(v))$. σ_Q induces a relation $\sqsubset_Q \subseteq \Delta \times \Gamma$ in the following way: if $\sigma_Q(u) = t$, then $v \sqsubset_Q u$ for all $v \,\epsilon\, \Delta$ occuring in t.
2. A *reduction ordering* $\lhd := (< \cup \sqsubset_Q)^+$ is the transitive closure of the tree ordering $<$ and the relation $\sqsubset_Q$. A first order substitution σ_Q is said to be *admissible* iff the reduction ordering $\lhd$ is irreflexive.

Example 5. $\sigma_Q = \{a_1 \backslash a, a_6 \backslash b\}$ is a first-order substitution for F_1. The induced reduction ordering is the tree ordering and irreflexive. Therefore σ_Q is admissible.

Theorem 6 (Characterization for Classical Logic [5]).
A (first-order) formula F is classically valid, iff there is a multiplicity $\mu := \mu_Q$, an admissible first-order substitution $\sigma := \sigma_Q$ and a set of σ-complementary connections such that every path through F^μ contains a connection from this set.

Example 6. Let μ_Q be the multiplicity of example 3 and $\sigma_Q = \{a_1 \backslash a, a_6 \backslash b\}$ be a first-order substitution. Then $\{\{a_2, a_4\}, \{a_6, a_5\}\}$ is a σ_Q-complementary set of connections and every path through F_1^μ contains a connection from this set. Therefore F_1 is classically valid.

[3] We consider substitutions to be *idempotent*, i.e. $\sigma(\sigma) = \sigma$ for a substitution σ.

2.3 Intuitionistic Logic

We shall now explain the extensions which are necessary to make the above characterization of validity applicable to intuitionistic logic. We define J-prefixes, intuitionistic substitutions and J-admissibility.

Definition 7 (J-Prefix). Let $u_1 < u_2 < \ldots < u_n = u$ be the elements of $\Psi \cup \Phi$ that dominate the atomic position u in the formula tree. The *J-prefix* $\mathrm{pre}_J(u)$ of u is defined as $\mathrm{pre}_J(u) := u_1 u_2 \ldots u_n$.

Example 7. For the formula F_1^μ we obtain $\mathrm{pre}_J(a_2) = a_0 A_1 A_2$, $\mathrm{pre}_J(a_7) = a_0 A_6 A_7$, $\mathrm{pre}_J(a_4) = a_0 a_4$ and $\mathrm{pre}_J(a_5) = a_0 a_5$.[4]

Definition 8 (Intuitionistic/Combined Substitution, J-Admissibility).

1. An *intuitionistic substitution* $\sigma_J : \Phi \to (\Phi \cup \Psi)^*$ assigns to each position of type ϕ a string over the alphabet $(\Phi \cup \Psi)$.
   An intuitionistic substitution σ_J induces a relation $\sqsubset_J \subseteq (\Phi \cup \Psi) \times \Phi$ in the following way: if $\sigma_J(u) = p$, then $v \sqsubset_J u$ for all characters v occuring in p.
2. A *combined substitution* $\sigma := (\sigma_Q, \sigma_J)$ consists of a first-order substitution σ_Q and an intuitionistic substitution σ_J. A connection $\{u, v\}$ is *σ-complementary* iff $\sigma_Q(\mathrm{lab}(u)) = \sigma_Q(\mathrm{lab}(v))$ and $\sigma_J(\mathrm{pre}_J(u)) = \sigma_J(\mathrm{pre}_J(v))$. The *reduction ordering* $\lhd := (< \cup \sqsubset_Q \cup \sqsubset_J)^+$ is the transitive closure of $<$, $\sqsubset_Q$ and $\sqsubset_J$.
3. A combined substitution $\sigma = (\sigma_Q, \sigma_J)$ is said to be *J-admissible* iff the reduction ordering $\lhd$ is irreflexive and the following condition holds for all $u \,\epsilon\, \Gamma$ and all $v \,\epsilon\, \Delta$ occuring in $\sigma_Q(u)$: $\sigma_J(\mathrm{pre}_J(u)) = \sigma_J(\mathrm{pre}_J(v)) q^*$ for some $q^* \,\epsilon\, (\Phi \cup \Psi)^*$.

Example 8. For the formula F_1^μ the combined substitution $\sigma = (\sigma_Q, \sigma_J)$ where $\sigma_Q = \{a_1 \backslash a, a_6 \backslash b\}$ and $\sigma_J = \{A_1 \backslash \varepsilon, A_2 \backslash a_4, A_6 \backslash \varepsilon, A_7 \backslash a_5\}$ is J-admissible.[5] Then $\{\{a_2, a_4\}, \{a_6, a_5\}\}$ is a set of σ-complementary connections.

Theorem 9 (Characterization for Intuitionistic Logic [16]).
A formula F is intuitionistically valid, iff there is a multiplicity $\mu := (\mu_Q, \mu_J)$, a J-admissible combined substitution $\sigma = (\sigma_Q, \sigma_J)$, and a set of σ-complementary connections such that every path through F^μ contains a connection from this set.

Example 9. For F_1 let $\mu = (\mu_Q, \mu_J)$ be a multiplicity, where $\mu_Q(a_1) = 2$ and $\mu_J(a_1) = \mu_J(a_2) = 1$, and σ the combined substitution of example 8. Concerning example 8 the formula F_1 is intuitionistically valid.

2.4 Modal Logics

We shall now describe the extensions which are necessary to make the characterization of validity applicable to the modal logics D, D4, S4, S5 and T. We will define M-prefixes, modal substitutions and $\mathcal{L}$-admissibility.

Definition 10 (M-Prefix). Let $u_1 < u_2 < \ldots < u_n < u$ be the elements of $\nu \cup \Pi$ that dominate the atomic position u in the formula tree. The *M-prefix* $\mathrm{pre}_M(u)$ of u is defined as $\mathrm{pre}_M(u) := u_1 u_2 \ldots u_n$ for the modal logics D, D4, S4 and T, and as $\mathrm{pre}_M(u) := u_n$ for S5 ($\mathrm{pre}_M(u) := \varepsilon$, if $n = 0$ for D, D4, S4, S5, and T).

Example 10. Consider $F_2 \equiv \Box \forall x P x \Rightarrow \Box \forall y \Diamond P y$ and its formula tree in figure 3. As M-prefixes we get $\mathrm{pre}_M(a_3) = A_1$ and $\mathrm{pre}_M(a_7) = a_4 A_6$ ($\mathrm{pre}_M(a_7) = A_6$ for S5).

---

[4] Positions of type ϕ (and ν) play the role of variables and are written in capital letters.
[5] By ε we denote the empty string.

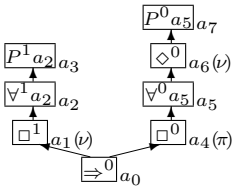

Fig. 3. Formula tree for F_2

domain \ $\mathcal{L}$	T	D	D4	S4	S5
·constant	–	–	–	–	–
·cumulative	$\sigma_M(\mathrm{pre}_M(u))=$ $\sigma_M(\mathrm{pre}_M(v))q$		$\sigma_M(\mathrm{pre}_M(u))=$ $\sigma_M(\mathrm{pre}_M(v))q^*$		–
·varying	$\sigma_M(\mathrm{pre}_M(u)) = \sigma_M(\mathrm{pre}_M(v))$				
accessibility	$\lvert\sigma_M(u)\rvert \leq 1$	$\lvert\sigma_M(u)\rvert = 1$	$\lvert\sigma_M(u)\rvert \geq 1$	–	–

Table 2. Domain/accessibility conditions for modal logics

Definition 11 (Modal/Combined Substitution, $\mathcal{L}$-Admissibility).

1. A *modal substitution* $\sigma_M : \nu \to (\nu \cup \Pi)^*$ induces a relation $\sqsubset_M \subseteq (\nu \cup \Pi) \times \nu$ as follows: if $\sigma_M(u)=p$, then $v \sqsubset_M u$ for all characters v occuring in p.
2. A connection $\{u, v\}$ is *σ-complementary*, where $\sigma := (\sigma_Q, \sigma_M)$ is a *combined substitution*, iff $\sigma_Q(\mathrm{lab}(u))=\sigma_Q(\mathrm{lab}(v))$ and $\sigma_M(\mathrm{pre}_M(u))=\sigma_M(\mathrm{pre}_M(v))$. The *reduction ordering* $\lhd$ is defined as $\lhd := (< \cup \sqsubset_Q \cup \sqsubset_M)^+$.
3. Let $\mathcal{L}$ be a modal logic such as D, D4, S4, S5 or T. A combined substitution $\sigma = (\sigma_Q, \sigma_M)$ is said to be *$\mathcal{L}$-admissible* iff the reduction ordering $\lhd$ is irreflexive, for all $u \in \Gamma$ the *domain condition* (for all $v \in \Gamma \cup \Delta$ occuring in $\sigma_Q(u)$, for some $q \in \nu \cup \Pi \cup \{\varepsilon\}$, $q^* \in (\nu \cup \Pi)^*$) and the *accessibility condition* holds (see table 2).

Example 11. Let $\sigma_i=(\sigma_Q, \sigma_{M\,i})$ for $i=1,2,3$ where $\sigma_Q=\{a_2 \backslash a_5\}$, $\sigma_{M\,1}=\{A_1 \backslash a_4 A_6\}$, $\sigma_{M2}=\{A_1 \backslash a_4, A_6 \backslash \varepsilon\}$ and $\sigma_{M3}=\{A_1 \backslash a_4, A_6 \backslash a_4\}$. The only path through F_2 consists of the connection $\{a_3, a_7\}$. This connection is σ_i-complementary in D, D4, S4, T (for $i=1,2$) and S5 (for $i=3$). σ_1 is D4- and S4-admissible for constant and cumulative domains, σ_2 is S4- and T-admissible for constant, cumulative and varying domains, σ_3 is S5-admissible for constant, cumulative and varying domains.

Theorem 12 (Characterization for Modal Logics [16]).
Let $\mathcal{L}$ be one of the modal logics D, D4, S4, S5 or T. A formula F is valid in the modal logic $\mathcal{L}$, iff there is a multiplicity $\mu := (\mu_Q, \mu_M)$, a $\mathcal{L}$-admissible combined substitution $\sigma = (\sigma_Q, \sigma_M)$, and a set of σ-complementary connections such that every path through F^μ contains a connection from this set.

Example 12. For F_2 let $\mu = (\mu_Q, \mu_M)$, where $\mu_Q(a_2) = \mu_M(a_1) = \mu_M(a_6) = 1$, and σ_i (for i=1,2,3) the combined substitutions as defined in example 11. The formula F_2 is valid in D4, S4, S5 and T for the constant and cumulative domain and valid in S4, S5 and T for the varying domain.

3 A Uniform Proof Search Procedure

According to the above matrix characterization the validity of a formula F can be proven by showing that all paths through the matrix representation of F contain a complementary connection. In this section we describe a general path checking algorithm which is driven by connections instead of the logical connectives. Once a complementary connection has been identified all paths containing this connection are deleted. This is similar to Bibel's connection method for classical logic [5] but without necessity for transforming the formula into any normal form. The algorithm avoids the notational redundancies occurring in sequent or analytic tableaux based proof procedures. Our technique can also be used for proof search in various non-classical logics where no normal form exists. Based on the matrix characterization we only have to modify the notion of complementarity while leaving the path checking algorithm unchanged.

3.1 Definitions and Notation

Before introducing the algorithm we specify some basic definitions and notations which allow a very short, elegant, and uniform presentation. We define α-/β-related positions, active goals, active paths, proven subgoals and provable subgoals.

Definition 13 (α-related, β-related).

1. Two positions u and v are *α-related*, denoted $u{\sim_\alpha}v$, iff $u{\neq}v$ and the greatest common ancestor of u and v, wrt. the tree ordering $<$, is of principal type α.
2. Two positions u and v are *β-related*, denoted $u{\sim_\beta}v$, iff $u{\neq}v$ and the greatest common ancestor of u and v, wrt. the tree ordering $<$, is of principal type β.

Remark. If two atoms are α-related they appear side by side in the matrix of a formula, whereas they appear one upon the other if they are β-related.

Example 13. Let $F_3 \equiv S \wedge (\neg(T{\Rightarrow}R){\Rightarrow}P){\Rightarrow}(\neg((P{\Rightarrow}Q) \wedge (T{\Rightarrow} R)){\Rightarrow}\neg\neg P \wedge S)$ with the matrix of F_3 in figure 4.[6] Then, e.g., $S^1{\sim_\alpha}T^0$, $R^1{\sim_\alpha}T^1$, $P^1{\sim_\beta}T^0$ or $T^1{\sim_\beta}Q^0$.

$$\left[\quad S^1 \quad \begin{bmatrix} T^0 \\ R^1 \\ P^1 \end{bmatrix} \quad \left[\begin{bmatrix} \tilde{P}^1 & Q^0 \end{bmatrix} \begin{bmatrix} T^1 & R^0 \end{bmatrix}\right] \quad \begin{bmatrix} S^0 \\ P^0 \end{bmatrix} \quad\right]$$

Fig. 4. Matrix of the formula F_3

Definition 14 (α-/β-related for a Set of Positions).

1. A position u and a set of positions $\mathcal{S}$ are *α-related* ($u{\sim_\alpha}\mathcal{S}$) if $u{\sim_\alpha}v$ for all $v{\epsilon}\mathcal{S}$.
2. A position u and a set of positions $\mathcal{S}$ are *β-related* ($u{\sim_\beta}\mathcal{S}$) if $u{\sim_\beta}v$ for all $v{\epsilon}\mathcal{S}$.

Example 14. For the F_3 we obtain, e.g., $T^1{\sim_\alpha}\{R^0,S^1,R^1\}$ and $T^0{\sim_\beta}\{R^1,P^1\}$.

The following definitions and lemmas always hold for a given (possibly indexed) formula F. By $\mathcal{A}$ we denote the set of all atomic positions in the formula F.

Definition 15 (Subpath, Subgoal).

1. A *subpath* $\mathcal{P}{\subseteq}\mathcal{A}$ is a set of atomic positions with $u{\sim_\alpha}(\mathcal{P}{\setminus}\{u\})$ for all $u\,{\epsilon}\,\mathcal{P}$.
2. A *subgoal* $\mathcal{C}{\subseteq}\mathcal{A}$ is a set of atomic positions with $u{\sim_\beta}(\mathcal{C}{\setminus}\{u\})$ for all $u\,{\epsilon}\,\mathcal{C}$.

Remark. A subpath is a, possible uncompleted, "horizontal path" through a matrix; a subgoal is a, possible uncompleted, "vertical path" through a matrix.

Example 15. For F_3, e.g., $\mathcal{P}_1{=}\{S^1, T^1, P^0\}, \mathcal{P}_2{=}\{S^1, R^1, S^0\}$ and $\mathcal{P}_3{=}\{T^0, T^1, R^0\}$ are subpaths, $\mathcal{C}_1{=}\{T^0\}, \mathcal{C}_2{=}\{Q^0, T^1\}$ and $\mathcal{C}_3{=}\{S^1\}$ are subgoals.

For a valid formula F every path through F has to contain a complementary connection (i.e. every path has to be complementary). The proof procedure described afterwards are based on the notion of "active path" and "proven subgoal".

Definition 16 (Active Goal, Active Path, Proven Subgoal).
An *active goal* is a tuple $(\mathcal{P}, \mathcal{C})$ consisting of a subpath $\mathcal{P}$ and a subgoal $\mathcal{C}$ with $u{\sim_\alpha}\mathcal{P}$ for all $u\,{\epsilon}\,\mathcal{C}$. We call $\mathcal{P}$ an *active path* and $\mathcal{C}$ a *proven subgoal*.

[6] In the following we use the labels instead of the (atomic) positions themselves. To distinguish the two atoms P^1 one atom is marked with a tilde (˜).

During proof search the active path will specify those paths which are just being investigated for complementarity. All paths which contain the active path $\mathcal{P}$ and additionally one element u of the proven subgoal will already have been proven complementary, since it has been shown that they contain a complementary connection (which may have occured within an extension of $\mathcal{P} \cup \{u\}$).

Example 16. For the formula F_3, e.g., $(\mathcal{P}_1, \mathcal{C}_1)$, $(\mathcal{P}_2, \mathcal{C}_2)$ and $(\mathcal{P}_3, \mathcal{C}_3)$ are active goals. $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$ are active paths, $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$ are proven subgoals.

Definition 17 (Open Subgoal). An *open subgoal* $\mathcal{C}' \subseteq \mathcal{A}$ with respect to an active goal $(\mathcal{P}, \mathcal{C})$ is a set of atomic positions, such that $u \sim_\alpha \mathcal{P}$ and $u \sim_\beta \mathcal{C}$ for all $u \in \mathcal{C}'$ and there is no $v \in \mathcal{A}$ with $v \notin \mathcal{C}'$, $v \sim_\alpha \mathcal{P}$ and $v \sim_\beta \mathcal{C}$.

An open subgoal together with the active path specifies paths which include the active path but still have to be tested for complementarity. If all paths including the active path and one element of the open subgoal are proved successfully for complementarity then all paths containing the active path are complementary. Note that there may be more than one open subgoal for a given active goal.

Example 17. For F_3 the set $\{R^1, P^1\}$ is an open subgoal wrt. the active goal $(\mathcal{P}_1, \mathcal{C}_1)$ (example 16); the empty set $\emptyset$ is the only subgoal wrt. $(\mathcal{P}_2, \mathcal{C}_2)$ or $(\mathcal{P}_3, \mathcal{C}_3)$.

Lemma 18 (Paths). *A subpath $\mathcal{P} \subseteq \mathcal{A}$ is a path iff there is no $u \in \mathcal{A}$ with $u \sim_\alpha \mathcal{P}$.*

Proof of the lemma. A path is a "complete horizontal path" through a matrix. □

Definition 19 (Provable Goal). An active goal $(\mathcal{P}, \mathcal{C})$ is called *provable* with respect to a formula F, iff there is an open subgoal $\mathcal{C}'$ wrt. $(\mathcal{P}, \mathcal{C})$, such that for all $u \in \mathcal{C}'$ all paths $\mathcal{P}'$ through F with $P \cup \{u\} \subseteq \mathcal{P}'$ are complementary.

Lemma 20 (Empty Open Subgoal). *Let $(\mathcal{P}, \mathcal{C})$ an active goal such that there is no $u \in \mathcal{A}$ with $u \sim_\alpha \mathcal{P}$ and $u \sim_\beta \mathcal{C}$. Then $(\mathcal{P}, \mathcal{C})$ is provable.*

Proof of the lemma. If there is no $u \in \mathcal{A}$ so that $u \sim_\alpha \mathcal{P}$ and $u \sim_\beta \mathcal{C}$, then $\mathcal{C}' = \emptyset$ is the only open subgoal wrt. $(\mathcal{P}, \mathcal{C})$ (definition 17). Since the open subgoal $\mathcal{C}'$ wrt. $(\mathcal{P}, \mathcal{C})$ is empty, the active goal $(\mathcal{P}, \mathcal{C})$ is provable (definition 19). □

Proposition 21 (Provable Goal). *Let $(\mathcal{P}, \mathcal{C})$ be an active goal and there is an open subgoal $\mathcal{C}'$ wrt. $(\mathcal{P}, \mathcal{C})$ with $\mathcal{C}' \neq \emptyset$. $(\mathcal{P}, \mathcal{C})$ is provable, iff there is a complementary connection $\{A, \bar{A}\}$ with $A \sim_\alpha \mathcal{P}$ and $A \sim_\beta \mathcal{C}$, such that*
1. $\bar{A} \in \mathcal{P}$ or
2. $\bar{A} \sim_\alpha (\mathcal{P} \cup \{A\})$ and the active goal $(\mathcal{P} \cup \{A\}, \{\bar{A}\})$ is provable
and the active goal $(\mathcal{P}, \mathcal{C} \cup \{A\})$ is provable.

Lemma 22 (Provability of $(\emptyset, \emptyset)$). *Let F be a formula. The active goal $(\emptyset, \emptyset)$ wrt. F is provable, iff every path through F is complementary.*

Proof of the lemma. $(\emptyset, \emptyset)$ is provable iff there is an open subgoal $\mathcal{C}' \neq \emptyset$ such that every path $\mathcal{P}'$ with $\{u\} \subseteq \mathcal{P}'$ and $u \in \mathcal{C}'$ is complementary. Since every path through F contains an $u \in \mathcal{C}'$ (lemma 18, definition 17) they are all complementary. □

Theorem 23 (Validity of a Formula).
A formula F is valid in classical logic (intuitionistic logic/modal logic $\mathcal{L}$) iff there is a multiplicity μ and an admissible ($\mathcal{J}$-admissible/$\mathcal{L}$-admissible combined) substitution σ such that the active goal $(\emptyset, \emptyset)$ wrt. F^μ is provable.

Proof of the theorem. Follows from lemma 22 and the characterizations of validity for classical, intuitionistic, and modal logics (theorems 6,9, and 12). □

3.2 The Algorithm

We now represent the proof procedure for classical ($\mathcal{C}$), intuitionistic ($\mathcal{J}$) and the modal logics D, D4, S4, S5 and T. It consists of the uniform path-checking algorithm and a complementarity test which will be described in the next section.

Let F be a (first-order) formula, μ a multiplicity, σ a substitution, and F^μ an indexed formula. Furthermore let $\mathcal{A}^\mu$ be the set of atomic positions in the formula tree for F^μ and $\sim_\alpha$ and $\sim_\beta$ the relations α- and β-related.

The main function $\textsc{Proof}_\mathcal{L}(F)$ in figure 5 gets an additional parameter $\mathcal{L}$ where $\mathcal{L} \in \{\mathcal{C}, \mathcal{J}, \text{D}, \text{D4}, \text{S4}, \text{S5}, \text{T}\}$ specifies the logic $\mathcal{L}$ under consideration. It returns *true* iff the formula F is valid in the corresponding logic.

```
Function PROOF_L(F)
    Input:    first-order formula F
    Output:   true, if, and only if, F is valid in the L-logic
begin PROOF_L;
    m := 0;
    repeat
        m := m+1;   INITIALIZE_L^(m)(μ, σ);
        valid := SUBPROOF_L(F^μ, ∅, ∅);
    until valid=true;
    return true;
end PROOF_L.
```

Fig. 5. Function $\textsc{Proof}_\mathcal{L}(F)$

The function $\textsc{Proof}_\mathcal{L}(F)$ first initializes the multiplicity μ and the (combined) substitution σ according to table 3. After that the function $\textsc{Subproof}_\mathcal{L}$ is invoked on the initial subgoal $(\emptyset, \emptyset)$. If it returns *false* the multiplicity is increased stepwisely, until the function Subproof returns *true*.[7]

$\mathcal{L}$	$\mathcal{C}$	$\mathcal{J}$	D, D4, S4, S5, T
multiplicity μ	$\mu(u){:=}m,\ \forall u \in \Gamma$	$\mu(u){:=}m,\ \forall u \in \Gamma \cup \Phi$	$\mu(u){:=}m,\ \forall u \in \Gamma \cup \nu$
substitution σ	$\sigma{:=}\emptyset$	$\sigma{:=}(\emptyset, \emptyset)$	$\sigma{:=}(\emptyset, \emptyset)$

Table 3. $\textsc{Initialize}_\mathcal{L}^{(m)}(\mu, \sigma)$

The function $\textsc{Subproof}_\mathcal{L}(F^\mu, \mathcal{P}, \mathcal{C})$ in figure 6 implements the test wether the active goal $(\mathcal{P}, \mathcal{C})$ is provable with respect to F^μ. Note that all variables except for the set of atomic positions $\mathcal{A}^\mu$ and the (combined) substitution σ are local.

Within $\textsc{Subproof}_\mathcal{L}$ a function $\textsc{Complementary}_\mathcal{L}(F^\mu, A, \bar{A}, \sigma', \sigma)$ is used, which returns true iff the connection $\{A, \bar{A}\}$ in the (indexed) formula F^μ is complementary under a (combined) substitution σ in the logic $\mathcal{L}$ taking into account the substitution σ' computed so far. In this case the substitution σ is returned. $\textsc{Complementary}_\mathcal{L}$ depends on the peculiarities of each logic (see section 4).

Lemma 24 (Correctness and Completeness of $\textsc{Subproof}_\mathcal{L}$).
The function $\textsc{Subproof}_\mathcal{L}(F^\mu, \mathcal{P}, \mathcal{C})$ *returns* true *if the active goal* $(\mathcal{P}, \mathcal{C})$ *is provable with respect to* F^μ*; otherwise it returns* false.

Proof of the lemma. The function $\textsc{Subproof}_\mathcal{L}$ uses the statements for a provable goal made in lemma 20 and proposition 21. □

[7] This technique of stepwise increasing the multiplicity is obviously not very efficient. In an efficient implementation the multiplicity for each suitable position is determined individually *during* the path-checking process (as in the usual connection method).

```
Function SUBPROOF_L(F^μ, P, C)
    Input:    indexed formula F^μ, active path P⊆A^μ, proven subgoals C⊆A^μ
    Output:   true, if (P, C) wrt. F^μ is provable; false, otherwise
begin SUBPROOF_L;
    if there is no A ∈ A^μ where A∼_α P and A∼_β C then return true;
    E := ∅;   σ' := σ;
    repeat
        select A ∈ A^μ where A∼_α(P∪E) and A∼_β C;
        if there is no such A then return false;
        E := E ∪ {A};   D := ∅;   valid := false;   noconnect := false;
        repeat
            select Ā ∈ A^μ where Ā ∉ D and [COMPLEMENTARY_L(F^μ, A, Ā, σ', σ)]
            and (1.) Ā ∈ P or (2.) Ā∼_α(P∪{A});
            if there is no such Ā
                then noconnect := true
                else  D := D ∪ {Ā}
                      if Ā ∈ P then valid := true
                                else valid := SUBPROOF_L(F^μ, P∪{A}, {Ā});
                      if valid=true then valid := SUBPROOF_L(F^μ, P, C∪{A});
        until valid=true or noconnect=true;
    until valid=true;
    return true;
end SUBPROOF_L.
```

Fig. 6. Function $\textsc{Subproof}_{\mathcal{L}}(F^{\mu}, \mathcal{P}, \mathcal{C})$

Theorem 25 (Correctness and Completeness of $\textsc{Proof}_{\mathcal{L}}$).
The function $\textsc{Proof}_{\mathcal{L}}(F)$ *returns* true*, iff the formula* F *is* $\mathcal{L}$*-valid.*

Proof of the theorem. $\textsc{Proof}_{\mathcal{L}}$ uses the relation between the validity of F wrt. logic $\mathcal{L}$ and the provability of the active goal $(\emptyset, \emptyset)$ wrt. F^{μ} (theorem 23) and searches for a suitable multiplicity μ. The theorem thus follows from lemma 24. □

Remark. Our algorithm is *not a decision procedure*, i.e. for an invalid formula it will not terminate. Note that the propositional part of each logic under consideration is decidable whereas in the first-order case there are no decision procedures.

4 Complementarity in Classical and Non-Classical Logics

Matrix characterizations of validity are based on the existence of admissible (combined) substitutions that render a set of simultaneously complementary connections such that every path contains a connection from this set. In our path-checking algorithm we have to ensure that after adding a connection to the current set there still is an admissible substitution under which *all* connections are complementary. For this purpose we invoke $\textsc{Complementary}_{\mathcal{L}}$ which implements a complementarity test for two atoms and depends on the particular logic. Whereas in the classical logic term-unification and a test for irreflexivity of the reduction ordering suffices to compute an admissible substitution, we need an additional string-unification algorithm to unify prefixes in the case of intuitionistic and modal logic.

After explaining the complementarity test for classical logic we extend it to intuitionistic logic and present details of the string-unification algorithm. We then deal with D, D4, S4, S5 and T which can be treated in a uniform way.

4.1 Classical Logic $\mathcal{C}$

In $\mathcal{C}$ two atoms are complementary if they can be unified under a term-substitution and the induced reduction ordering is irreflexive. For the former well-known unification algorithms [14, 9] compute a most general unifier which leads to the first-order substitution. For the latter efficient algorithms testing the acyclicity of a directed graph can be used. COMPLEMENTARY$_{\mathcal{C}}$ succeeds and returns σ_Q iff the two conditions in table 4 hold ($\texttt{term\_unify}(s,t)$ expresses term unification of s and t and results together with ${\sigma_Q}'$ in the substitution σ_Q).

	$\mathcal{C}$
term unification	$\texttt{term\_unify}({\sigma_Q}'(\text{lab}(A)), {\sigma_Q}'(\text{lab}(\bar{A}))) \rightsquigarrow \sigma_Q$
reduction ordering	$\lhd := (< \cup \sqsubset_Q)^+$ is irreflexive

Table 4. Function COMPLEMENTARY$_{\mathcal{C}}(F^\mu, A, \bar{A}, {\sigma_Q}', \sigma_Q)$

4.2 Intuitionistic Logic $\mathcal{J}$

For intuitionistic logic the complementarity test additionally requires that the prefixes of two connected atoms can be unified. For this purpose we use a specialized string unification (T-String-Unification) $\texttt{T\_string\_unify}_J(p,q)$, described below, which succeeds if the prefixes p and q can be unified and leads altogether the substitution σ_J. To guarantee admissibility the extended reduction ordering has to be irreflexive and an 'additional condition' has to be fulfilled. If all the properties of table 5 hold the function COMPLEMENTARY$_{\mathcal{J}}$ succeeds and returns (σ_Q, σ_J).

	$\mathcal{J}$
term unification	$\texttt{term\_unify}({\sigma_Q}'(\text{lab}(A)), {\sigma_Q}'(\text{lab}(\bar{A}))) \rightsquigarrow \sigma_Q$
string unification	$\texttt{T\_string\_unify}_J({\sigma_J}'(\text{pre}_J(A)), {\sigma_J}'(\text{pre}_J(\bar{A}))) \rightsquigarrow \sigma_J$
additional condition*	$\lvert\sigma_J(\text{pre}_J(v))\rvert \leq \lvert\sigma_J(\text{pre}_J(u))\rvert$
reduction ordering	$\lhd := (< \cup \sqsubset_Q \cup \sqsubset_J)^+$ is irreflexive ($\sqsubset_J$ induced by $\sigma_\varepsilon^{(J)} \circ \sigma_J$)

$\sigma_\varepsilon^{(J)}(w) := \varepsilon$ for all $w \in \Phi$, $\circ$ is the composition of substitutions;

* must hold for all $u \in \Gamma$ and all $v \in \Delta$ occuring in $\sigma_Q(u)$.

Table 5. Function COMPLEMENTARY$_{\mathcal{J}}(F^\mu, A, \bar{A}, ({\sigma_Q}', {\sigma_J}'), (\sigma_Q, \sigma_J))$

T-String-Unification To unify the prefixes of two connected atoms we use a specialized string unification which respects the restrictions on the prefix strings p and q: no character is repeated either in p nor in q and equal characters only occur within a common substring at the beginning of p and q. This restriction allows us to give an efficient algorithm computing a minimal set of most general unifiers. Similar to the ideas of Martelli and Montanari [9] rather than by giving a recursive procedure we consider the process of unification as a sequence of transformations of an equation.

We start with a given equation $\Gamma = \{p{=}q\}$ and an empty substitution $\sigma = \emptyset$ and stop with an empty set $\Gamma = \emptyset$ and a substitution σ representing an idempotent most general unifier. Each transformation step replaces a tuple Γ, σ by a modified tuple Γ', σ'. The algorithm is described by transformation rules which can be applied nondeterministically. The set of most general unifiers consists of the results of all successfully finished transformations. For technical reasons we divide the right part q of the equation into two parts $q_1|q_2$, i.e. we start with $\{p = q_1|q_2\}, \{\}$.

Definition 26 (Transformation Rules for $\mathcal{J}$).
Let $\mathcal{V}$ be a set of variables, $\mathcal{C}$ a set of constants, and $\mathcal{V}'$ a set of *auxiliary variables* with $\mathcal{V} \cap \mathcal{V}' = \emptyset$. The set of *transformation rules* for $\mathcal{J}$ is defined in table 6.

R1.	$\{\varepsilon = \varepsilon \vert \varepsilon\},\ \sigma$	$\rightarrow$	$\{\},\ \sigma$
R2.	$\{\varepsilon = \varepsilon \vert t^+\},\ \sigma$	$\rightarrow$	$\{t^+ = \varepsilon \vert \varepsilon\},\ \sigma$
R3.	$\{Xs = \varepsilon \vert Xt\},\ \sigma$	$\rightarrow$	$\{s = \varepsilon \vert t\},\ \sigma$
R4.	$\{Cs = \varepsilon \vert Vt\},\ \sigma$	$\rightarrow$	$\{Vt = \varepsilon \vert Cs\},\ \sigma$
R5.	$\{Vs = z \vert \varepsilon\},\ \sigma$	$\rightarrow$	$\{s = \varepsilon \vert \varepsilon\},\ \{V \backslash z\} \cup \sigma$
R6.	$\{Vs = \varepsilon \vert C_1 t\},\ \sigma$	$\rightarrow$	$\{s = \varepsilon \vert C_1 t\},\ \{V \backslash \varepsilon\} \cup \sigma$
R7.	$\{Vs = z \vert C_1 C_2 t\},\ \sigma$	$\rightarrow$	$\{s = \varepsilon \vert C_2 t\},\ \{V \backslash z C_1\} \cup \sigma$
R8.	$\{Vs^+ = \varepsilon \vert V_1 t\},\ \sigma$	$\rightarrow$	$\{V_1 t = V \vert s^+\},\ \sigma$
R9.	$\{Vs^+ = z^+ \vert V_1 t\},\ \sigma$	$\rightarrow$	$\{V_1 t = V' \vert s^+\},\ \{V \backslash z^+ V'\} \cup \sigma$
R10.	$\{Vs = z \vert Xt\},\ \sigma$	$\rightarrow$	$\{Vs = zX \vert t\},\ \sigma$ $\quad (V \neq X$, and $s = \varepsilon$ or $t \neq \varepsilon$ or $X \in \mathcal{C})$

s, t and z denote (arbitrary) strings and s^+, t^+, z^+ a non-empty string. X, V, V_1, C, C_1 and C_2 denote single characters with $X \in \mathcal{V} \cup \mathcal{C} \cup \mathcal{V}'$, $V, V_1 \in \mathcal{V} \cup \mathcal{V}'$ (with $V \neq V_1$), and $C, C_1, C_2 \in \mathcal{C}$. $V' \in \mathcal{V}'$ is a new variable which does not occur in the substitution σ computed so far.

Table 6. Transformation Rules for Intuitionistic Logic (and Modal Logic S4)

By modifying the set of transformation rules we are also able to treat the modal logics D, D4, S5 and T (see [12] for details). Our algorithm is much simpler and considerably more efficient than other string unification algorithms developed so far. Ohlbach's algorithm [10], e.g., does not compute a minimal set of unifiers. In general the number of most general unifiers is finite but may grow exponentially with respect to the length of the unified strings.

4.3 Modal Logics

The modal logics also require T-string-unification. We have developed such procedures in [12] by considering the accessibility condition for each particular logic $\mathcal{L}$.[8] $\texttt{T\_string\_unify}_{\mathcal{L}}(p, q)$ computes the most general unifier for p and q with respect to the accessibility condition for $\mathcal{L}$. This leads to the substitution σ_M. For the cumulative and varying domain variants also the domain condition has to be taken into account. In table 7 we have summarized all conditions that have to be tested. $\textsc{Complementary}_{\mathcal{L}}$ succeeds, if all conditions in the row for $\mathcal{L}$ hold.

$\mathcal{L}$	D	D4	S4	S5	T
term unification	$\texttt{term\_unify}(\sigma_Q{}'(\mathrm{lab}(A)), \sigma_Q{}'(\mathrm{lab}(\bar{A}))) \rightsquigarrow \sigma_Q$				
string unification	$\texttt{T\_string\_unify}_{\mathcal{L}}(\sigma_M{}'(\mathrm{pre}_M(A)), \sigma_M{}'(\mathrm{pre}_M(\bar{A}))) \rightsquigarrow \sigma_M$				
domain condition					
· constant	–	× [9]	–	–	–
· cumulative*	$\lvert V\rvert \leq \lvert U\rvert \leq \lvert V\rvert+1$	$\lvert V\rvert \leq \lvert U\rvert$	$\lvert V\rvert \leq \lvert U\rvert$	–	$\lvert V\rvert \leq \lvert U\rvert \leq \lvert V\rvert+1$
· varying*	$\lvert U'\rvert = \lvert V'\rvert$	$\lvert U\rvert = \lvert V\rvert$	$\lvert U'\rvert = \lvert V'\rvert$	$\texttt{uni}_{S5} \rightsquigarrow \sigma_M$	$\lvert U'\rvert = \lvert V'\rvert$
reduction ordering	$\lhd := (< \cup \sqsubset_Q \sqsubset_M)^+$ is irreflexive $(\sqsubset_M$ induced by $\sigma_M^*)$				

$U := \sigma_M(\mathrm{pre}_M(u))$, $V := \sigma_M(\mathrm{pre}_M(v))$, $U' := \sigma_\varepsilon^{(M)} \circ \sigma_M(\mathrm{pre}_M(u))$, $V' := \sigma_\varepsilon^{(M)} \circ \sigma_M(\mathrm{pre}_M(v))$, $\sigma_\varepsilon^{(M)}(w) := \varepsilon$ for all $w \in \mathcal{V}$; * must hold for all $v \in \Pi$ ($v \in \mathcal{V} \cup \Pi$ for varying domains) occuring in $\sigma_Q(u)$; $\texttt{uni}_{S5} \cong \texttt{T\_string\_unify}_{S5}(\sigma_M(\mathrm{pre}_M(u)), \sigma_M(\mathrm{pre}_M(v)))$; $\sigma_M^* := \sigma_\varepsilon^{(M)} \flat \sigma_M$ for D,S4,S5,T / σ_M for D4.

Table 7. Function $\textsc{Complementary}_{\mathcal{L}}(F^\mu, A, \bar{A}, (\sigma_Q{}', \sigma_M{}'), (\sigma_Q, \sigma_M))$

[8] For S4 the algorithm for intuitionistic logic ($\texttt{T\_string\_unify}_J$) suffices.

[9] We do *not* deal with the constant domain of D4 since there is need for additional search when computing the modal substitution σ_M.

5 Conclusion

We have presented a uniform proof procedure for classical and some non-classical logics which is based on a unified representation of matrix characterizations of logical validity for these logics. It consists of a connection-driven general path-checking algorithm for arbitrary formulae and a component for checking the complementarity of two atomic formulae according to the peculiarities of a particular logic. By presenting appropriate components for classical logic, intuitionistic logic, and the modal logics D, D4, S4, S5, and T we have demonstrated that our procedure is suited to deal with a rich variety of logics in a simple and efficient way.

The algorithm has been implemented in Prolog in order to allow practical experiments. In the future we intend to elaborate optimizations like a decision procedure for the propositional case, efficiency improvements like a dynamic increase of the multiplicities instead of a global one, and to provide a C-implementatuion to allow realistic comparisons. Furthermore we intend to investigate extensions of our procedure to additional logics such as (fragments of) linear logic and other non-classical logics for which a matrix characterization can be developed.

References

1. B. Beckert and J. Posegga. leanTAP: lean, tableau-based theorem proving. *Proc. CADE-12*, LNAI 814, 1994.
2. E. W. Beth. *The foundations of mathematics.* North–Holland, 1959.
3. W. Bibel, S. Brüning, U. Egly, T. Rath. Komet. *Proc. CADE-12*, LNAI 814, pp. 783–787. 1994.
4. W. Bibel. On matrices with connections. *Jour. of the ACM*, 28, p. 633–645, 1981.
5. W. Bibel. *Automated Theorem Proving.* Vieweg, 1987.
6. M. C. Fitting. *Intuitionistic logic, model theory and forcing.* North–Holland, 1969.
7. G. Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1935.
8. R. Letz, J. Schumann, S. Bayerl, W. Bibel. Setheo: A high-performance theorem prover. *Journal of Automated Reasoning*, 8:183–212, 1992.
9. A. Martelli and Ugo Montanari. An efficient unification algorithm. *ACM TOPLAS*, 4:258–282, 1982.
10. H. J. Ohlbach. A resolution calculus for modal logics. Ph.D. Thesis, Universität Kaiserslautern, 1988.
11. J. Otten, C. Kreitz. A connection based proof method for intuitionistic logic. *Proc. 4th TABLEAUX Workshop*, LNAI 918, pp. 122–137, 1995.
12. J. Otten, C. Kreitz. T-String-Unification: Unifying Prefixes in Non-Classical Proof Methods. *Proc. 5th TABLEAUX Workshop*, LNAI 1071, pp. 244–260, 1996.
13. J. Otten. Ein konnektionenorientiertes Beweisverfahren für intuitionistische Logik. Master's thesis, TH Darmstadt, 1995.
14. J. A. Robinson. A machine-oriented logic based on the resolution principle. *Jour. of the ACM*, 12(1):23–41, 1965.
15. L. Wallen. Matrix proof methods for modal logics. *IJCAI-87* , p. 917–923. 1987.
16. L. Wallen. *Automated deduction in nonclassical logic.* MIT Press, 1990.
17. L. Wos et. al. Automated reasoning contributes to mathematics and logic. *Proc. CADE-10*, LNCS 449, pp. 485–499, 1990.

This article was processed using the LaTeX macro package with LLNCS style