

Electronic Notes in Theoretical Computer Science 262 (2010) 17–32

[www.elsevier.com/locate/entcs](http://www.elsevier.com/locate/entcs)

A Tableau Method for Checking Rule Admissibility in S4

Sergey Babenyshev[a](#_bookmark0), Vladimir Rybakov[a](#_bookmark0), Renate A. Schmidt[b](#_bookmark1) and Dmitry Tishkovsky[b](#_bookmark1)

a *Department of Computing and Mathematics,*

*Manchester Metropolitan University, UK*

b *School of Computer Science, The University of Manchester, UK*

Abstract

Rules that are admissible can be used in any derivations in any axiomatic system of a logic. In this paper we introduce a method for checking the admissibility of rules in the modal logic S4. Our method is based on a standard semantic ground tableau approach. In particular, we reduce rule admissibility in S4 to satisfiability of a formula in a logic that extends S4. The extended logic is characterised by a class of models that satisfy a variant of the co-cover property. The class of models can be formalised by a well-defined first- order specification. Using a recently introduced framework for synthesising tableau decision procedures this can be turned into a sound, complete and terminating tableau calculus for the extended logic, and gives a tableau-based method for determining the admissibility of rules.

*Keywords:* Tableau calculus, admissible rule, modal logic, S4, tableau synthesis framework.

# Introduction

Logical admissible rules were first considered by Lorenzen [[12](#_bookmark81)]. Initial investigations were limited to observations on the existence of interesting examples of admissible rules that are not derivable (see Harrop [[7](#_bookmark74)], Mints [[13](#_bookmark82)]). The area gained new momentum when Friedman [[2](#_bookmark70)] posed the question whether algorithms exist for re- cognising whether rules in intuitionistic propositional logic IPC are admissible. This problem and the corresponding problem for modal logic S4 are solved affirmatively in Rybakov [[15](#_bookmark84),[17](#_bookmark86)]. The same approach can be used for a broad range of proposi- tional modal logics, for example K4, S4, GL [[18](#_bookmark87)]. Roziere [[14](#_bookmark83)] presents a solution to Friedman’s problem for IPC that uses methods of proof theory.

The theory of admissible rules in S4 does not have the finite approximation property [[11](#_bookmark80)] in the strict sense. The algorithm in [[15](#_bookmark84)] is based on the existence of a model (of bounded size) that witnesses the non-admissibility of a rule, but is not

1571-0661 © 2010 Elsevier B.V. Open access under [CC BY-NC-ND license.](http://creativecommons.org/licenses/by-nc-nd/3.0/)

doi:10.1016/j.entcs.2010.04.003

necessarily a model for all the other admissible rules. In [[1](#_bookmark71)] it is observed that a witnessing model can be obtained by filtration.

Admissibility of rules has direct connections to the problem of unification. A refined technique [[18](#_bookmark87)] is used for admissibility of rules with meta-variables, for uni- fication, for unification with parameters, and for the solvability of logical equations in transitive modal logics.

Algorithms deciding admissibility for some transitive modal logics and IPC,

based on projective formulae and unification, are described in Ghilardi [[3](#_bookmark72),[4](#_bookmark73),[5](#_bookmark75),[6](#_bookmark76)]. They combine resolution and tableau approaches for finding projective approxima- tions of a formula and rely on the existence of an algorithm for theorem proving. A

practically feasible realisation for S4 built on the algorithm for IPC in [[5](#_bookmark75)] is described

in [[24](#_bookmark93)]. These algorithms were specifically designed for finding general solutions for matching and unification problems. In contrast, the original algorithm of [[15](#_bookmark84)] can be used to find only *some* solution of such problems in S4.

In this paper we focus on S4 and introduce a new method for recognising the admissibility of rules. Our method is based on the reduction of the problem of rule admissibility in S4 to the satisfiability of a certain formula in an extension of S4. We refer to the extended logic as S4u,+. S4u,+ can be characterised by a class of models, which satisfy a variant of the co-cover property that is definable by modal formulae. This property is expressible in first-order logic and means that the semantics of the logic can be formalised in first-order logic. We exploit this property in order to devise a tableau calculus for S4u,+ in a recently introduced framework for automatically synthesising tableau calculi and decision procedures [[21](#_bookmark90),[22](#_bookmark91)].

The tableau synthesis method of [[22](#_bookmark91)] works as follows.

* 1. The user defines the formal semantics of the given logic in a many-sorted first-order language so that certain well-definedness conditions hold.
  2. The method automatically reduces the semantic specification of the logic to Skolemised implicational forms which are then rewritten as tableau inference rules. These are combined with some default closure and equality rules.

The set of rules obtained in this way provides a sound and constructively complete calculus. Furthermore, this set of rules automatically has a subformula property with respect to a finite subformula operator. If the logic can be shown to admit finite filtration with respect to a well-defined first-order semantics then adding a general blocking mechanism produces a terminating tableau calculus [[21](#_bookmark90)].

We show how, using this method, a sound, complete and terminating tableau calculus can be synthesised for the extended logic S4u,+. This tableau calculus is then incorporated into a method for solving the rule admissibility problem in S4.

The paper is structured as follows. In Section [2](#_bookmark3) we define the syntax and se- mantics of the modal logic S4, and its extension S4u with the universal modality, in such a way that they can be accommodated in the tableau synthesis framework of [[22](#_bookmark91)]. The section also defines standard modal logic constructions and notions re- quired for the main results of the paper. In Section [3](#_bookmark5), we give definitions of derivable and admissible rules for S4 and state Rybakov’s criterion for testing admissibility

|  |  |
| --- | --- |
| Definitions of connectives | Background theory |
| *∀x* *ν*(*¬p, x*) *↔ ¬ν*(*p, x*)  *∀x* *ν*(*p ∨ q, x*) *↔ ν*(*p, x*) *∨ ν*(*q, x*)  *∀x* *ν*( *p, x*) *↔ ∃y* (*R*(*x, y*) *∧ ν*(*p, y*)) | *∀x, y, z* *R*(*x, y*) *∧ R*(*y, z*) *→ R*(*x, z*)  *∀x R*(*x, x*) |

Figure 1. Specification of the semantics of S4 in *S*S4

of rules in S4. The reduction of rule admissibility in S4 to satisfiability in S4u,+ is described in Section [4](#_bookmark15). In Section [5](#_bookmark18), we show how the formulaic variant of the co-cover property can be expressed by a set of first-order formulae that provide a suitable background theory for the specification of the semantics of S4u,+ in the tableau synthesis framework. Within this framework we can then synthesise sound and complete tableau calculi for the logics S4u,+ and S4u. Under certain conditions it is possible to refine the rules of the calculus that are generated by default [[22](#_bookmark91)]. These conditions are true for the logics we consider and is discussed in Section [6](#_bookmark21). Section [7](#_bookmark24) describes how terminating tableau calculi can be obtained by adding the unrestricted blocking mechanism of [[20](#_bookmark89),[21](#_bookmark90)]. In Section [8](#_bookmark27), we finally give an al- gorithm for testing rule admissibility, but also rule derivability, in S4. In Section [9](#_bookmark69) we conclude with a discussion of the applicability of the algorithm and method to other logics and various problems closely related to admissibility.

For lack of space some of the details of the tableau synthesis framework are omitted; for these the interested reader is referred to [[22](#_bookmark91)] and also [[20](#_bookmark89),[21](#_bookmark90)].

# Syntax and Semantics

We denote the language of the modal logic S4 by *L*S4. *L*S4 is given by the standard modal language over a countable set of propositional variables *{p, q, p*0*, q*0*,.. .}*, the Boolean logical connectives *¬* and *∨*, and the modal connective. Other Boolean connectives such as *T*, *⊥*, *∧*, *→*, and *↔* and the modal connectiveare defined via

*¬*, *∨*, andas usual.

Let *L*u denote the extension of the language *L*S4 with the ‘somewhere’ modality

S4

*⟨*u*⟩*. The dual modality *¬⟨*u*⟩¬* is the universal modality, which is denoted by [u].

For a formula *φ*, we denote by sub(*φ*) the set of all subformulae of *φ*. Let

sub(Σ)

=def

*{*sub(*φ*) *| φ ∈* Σ*}* for any set of formulae Σ. We usually write

sub(*φ*1*,..., φn*) rather than sub(*{φ*1*,..., φn}*). A set of formulae Σ is called a *sig- nature* iff sub(Σ) = Σ.

Following the tableau synthesis framework in [[22](#_bookmark91)], we accommodate *L*u in a

S4

multi-sorted first-order specification language. This specification language includes a countable set *{x, y, z, x*0*, y*0*, z*0*,.. .}* of first-order variables that represent elements in models, the binary predicate symbol *R* of the background theory that represents

the accessibility relation, and a binary (intersort) predicate symbol *ν* which rep- resents the forcing relation *|*=. *ν* can be thought of as a holds predicate. Figure [1](#_bookmark2)

gives the first-order semantic specification of the standard semantics of S4 in the framework. We denote the specification by *S*S4. In addition to the formulae of Figure [1](#_bookmark2), *S*S4 includes the usual congruence axioms for the equality predicate *≈*, see [[22](#_bookmark91)] for details.

Thus, an S4*-(Kripke) model* (of a signature Σ) is a first-order model *M* =

*⟨WM , RM , νM ⟩* that validates all the formulae of Figure [1](#_bookmark2) under arbitrary substi- tutions of *L*S4-formulae (from the signature Σ) for variables *p* and *q*.

Let S4u be the extension of S4 with the somewhere (or universal) modality. The semantic specification of S4u, denoted by *S*S4u , is the extension of the specification *S*S4 of S4 with

(1) *∀x* (*ν*(*⟨*u*⟩p, x*) *↔ ∃y ν*(*p, y*))*.*

This defines the somewhere modality *⟨*u*⟩*. An S4u-model (of a signature Σ) is by definition a first-order model *M* = *⟨WM , RM , νM ⟩* that validates this formula in addition to all the formulae of Figure [1](#_bookmark2) under arbitrary substitutions of *L*u -formulae

S4

(from the signature Σ) for propositional variables *p* and *q*.

Note that if *M* is a model of the signature Σ then *M* is also a model of any signature Σ*' ⊆* Σ. In the other direction, it is clear that every model *M* of a smaller signature Σ*'* can be extended to a model of a bigger signature Σ by (re)defining the interpretation of *ν* on formulae from Σ *\* Σ*'* (by induction on their length). We often use these facts when defining models.

A *(Kripke) frame* of S4 (resp. S4u) is a first-order structure *F* = *⟨WF , RF ⟩* that validates all the formulae of the background theory of *S*S4 (resp. *S*S4u ). A model *M* is *based on a frame F* (or the *underlying frame* of *M* is *F* ) iff *WM* = *WF* and *RM* = *RF* .

A *cluster* of a model *M* is a set *U ⊆ WM* such that for all *w ∈ WM* and *u ∈ U* it holds that (*w, u*)*,* (*u, w*) *∈ RM* iff *w ∈ U* . Any set of *RM* -incomparable clusters of *M* is called an *anti-chain* in *M* . A cluster *U ⊆ WM* is *maximal* iff (*u, w*) *∈ RM* implies *w ∈ U* for all *w ∈ WM* and *u ∈ U* .

Let Σ be a fixed signature and *M* a model of the signature Σ. We say that a formula *φ ∈* Σ is *true* (*satisﬁed* ) in a world *w ∈ W M* (in symbols *M, w |*= *φ*) iff (*φ, w*) *∈ νM* . A formula *φ* (from Σ) is *valid* in *M* (in symbols *M |*= *φ*) iff *M, w |*= *φ* for every *w ∈ WM* . And, as usual, *φ* is *satisﬁable* in *M* iff there is *w ∈ WM* such that *M, w |*= *φ*. A formula *φ ∈* Σ is *valid* in a frame *F* (in symbols *F |*= *φ*) iff it is true in every model *M* of the signature Σ based on *F* .

For a signature Σ and every element *w* of *WM* we define a Σ*-type τ* Σ(*w*) of *w*

as the set of all formulae from Σ which are true in *w*, namely:

*τ* Σ(*w*)

=def

*{ψ ∈* Σ *| M, w |*= *ψ}.*

We omit the superscript Σ and write *τ* (*w*) if Σ is known from the context. A model *M* of a signature Σ is called Σ*-differentiated* iff *τ* Σ(*w*) = *τ* Σ(*v*) implies *w* = *v* for all *w, v ∈ WM* . A model *M* is *formulaic* iff for every element *w ∈ WM* there is a formula *φ* such that *M, w |*= *φ* and *M, v |*= *φ* for all *v ∈ WM \ {w}*. It is clear that

if Σ is finite then every Σ-differentiated model is also formulaic.

We call a Kripke model *M* of a signature Σ*'* a *deﬁnable variant* of a model *N* of a signature Σ if *M* and *N* are based on the same frame and for every propositional variable *p ∈* Σ*'* there is a formula *φ ∈* Σ such that *M, w |*= *p ⇐⇒ N, w |*= *φ* for every *w ∈ WM* = *WN* .

Let Σ be the set of all formulae in *n* variables *p*1*,..., pn*. An S4-model *M* of signature Σ is called an *n-characterising* model for S4 iff *M |*= *φ ⇐⇒ φ ∈* S4 for every *φ ∈* Σ.

# Rules Admissible for S4

Since the language of S4 contains conjunction, without loss of generality we consider only one-premise rules. A *rule* is a pair *⟨α, β⟩* of *L*S4-formulae, usually written *α/β*. A rule *r* = *α/β* is *valid* in a model *M* (written *M |*= *r*) iff *M |*= *α* implies *M |*= *β*. A rule *r* is *valid* on a frame *F* (written *F |*= *r*) iff *r* is valid in any model *M* based on *F* . Two rules *r*1, *r*2 are *semantically equivalent*, or simply *equivalent*, if *F |*= *r*1 *⇐⇒ F |*= *r*2 for any frame *F* .

A rule *r* = *α/β* is *derivable* in S4 iff *β* is derivable from *α* and the theorems of S4 with the rule of modus ponens and the rule of necessitation. It is clear that *r* is derivable in S4 iff *r* is valid in every S4-model. The following variant of the deduction theorem in S4 can be proved by standard methods.

**Theorem 3.1** *A rule α/β is derivable in* S4 *iff* [u]*α ∧ ¬β is unsatisﬁable in* S4*u.*

A rule *r* = *α/β* is *admissible* for the modal logic S4, written *r ∈* Adm(S4), if for every substitution *σ* from *σ*(*α*) *∈* S4 it follows that *σ*(*β*) *∈* S4.

A series Ch*n*(S4), *n >* 0, of formulaic and n-characterising S4 models is described in [[18](#_bookmark87)]. These models are used in the description of the following admissibility criterion.

**Theorem 3.2 (Corollary of Theorem 3.3.3 [**[**18**](#_bookmark87)**])** *A rule is admissible in* S4 *iff it is valid in a deﬁnable variant of* Ch*n*(S4) *for each n >* 0*.*

The most important property for the result of this paper is that the models Ch*n*(S4), *n >* 0, and their definable variants possess the co-cover property. That the admissible rules for S4 can be characterized by the class of models with the co-cover property is shown in [[16](#_bookmark85)]. A more general result appears in [[18](#_bookmark87), Theorem 3.5.1]. Similar characterisation results for IPC appear in [[9](#_bookmark77), Theorem 4.1(iv)] and [[10](#_bookmark78), Corollary 3.14].

By definition, a model *M* has the *co-cover property* (CCP) if for every finite anti- chain *D ⊆ WM* (*D* may be empty), there exists a one-element cluster *{w}⊆ WM* such that

*{u ∈ WM |* (*w, u*) *∈ RM }* = *{w}∪* *{u ∈ WM |* (*v, u*) *∈ RM }.*

*v∈D*

The one-element cluster is called a *co-cover for D*. Note that a co-cover for the empty anti-chain is a maximal one-element cluster of *M* .

A rule *r* is said to be in *reduced normal form* if it has the form

(rnf) *r* =

1≤*j*≤*s*

*φj* */p*0*,*

and each disjunct *φj* has the form

*φj* =

0≤*i*≤*n*

*pt*(*i,j,*0) *∧*

0≤*i*≤*n*

*i*

( *pi*)*t*(*i,j,*1)*,*

where (i) all *φj* are different, (ii) *p*0*,..., pn* denote propositional variables, (iii) *t* is a Boolean function *t* : *{*0*,..., n}× {*1*,..., s}× {*0*,* 1*}→ {*0*,* 1*}* (i.e., *t*(*i, j, z*) *∈ {*0*,* 1*}*), and (iv) *α*0 d=ef *¬α* and *α*1 =def *α* for any formula *α*.

Using the renaming technique any modal rule can be transformed into an equi- valent rule in reduced normal form [[15](#_bookmark84)].

**Lemma 3.3** *Any rule r* = *α/β can be transformed in exponential time to an equi- valent rule in reduced normal form.*

**Proof** We describe the algorithm of [[18](#_bookmark87)] (Lemma 3.1.3 and Theorem 3.1.11). Let *r* = *α/β* be a rule. We need a set of new variables *{qγ | γ ∈* sub(*α, β*)*}*. The first step is to replace *r* = *α/β* with *r*1 = *α ∧* (*qβ ↔ β*)*/qβ*. It is easy to see that *r* is refuted on a frame *F* iff *r*1 can be refuted on the same frame *F* . Therefore *r* and *r*1 are equivalent.

Inductive step: Suppose the rule *ri* = *γi/qβ* was obtained in the *i*th step. We call a formula *δ ∈* sub(*γi*) *∩* sub(*α, β*) *ﬁnal*, if it is not a variable and not a proper subformula of any other formula in sub(*γi*) *∩* sub(*α, β*). Let *Ti* be the set of all final formulae obtained at the *i*th step. We replace the rule *ri* with a new one, namely *ri*+1 = *γi*+1*/qβ*, where

*γi*+1 = *ti*(*γi*) *∧*

*qγ ∨qδ∈Ti*

((*qγ ↔ γ*) *∧* (*qδ ↔ δ*)) *∧*

*¬qδ, qδ∈Ti*

(*qδ ↔ δ*)*,*

and *ti*(*γi*) is the formula obtained from *γi* by replacing all final subformulae *δ* with

*qδ*. It is straightforward to check that *ri* and *ri*+1 are equivalent.

Note that every inductive step reduces the maximal height of the non-Boolean subformulae of the rule. Therefore after a finite number of steps we get a premise *γk*, which is a Boolean combination of *literals* of the form *p* or *p*, where *p* is a propos- itional variable. We denote this intermediate form by do(*r*) (depth-one form).

Finally, we transform the premise of the obtained rule *rN* = *γk/qβ* into disjunct- ive normal form over literals. This requires no more than exponential time on the number of variables, i.e., on the number of subformulae of the original rule, which is the same as for the reduction of any Boolean formula to disjunctive normal form.

The reduced normal form, obtained in Lemma [3.3](#_bookmark8), is uniquely defined and is denoted by rnf(*r*). Note that Lemma [3.3](#_bookmark8) proves more than the equivalence of *r* and rnf(*r*). In particular, from the proof it follows that if *r* is refutable in a model *N* then rnf(*r*) is refutable in a definable variant *M* of *N* with *M, w |*= *qγ ⇐⇒ N, w |*= *γ* for all *γ ∈* sub(*α, β*).

Let *r* be any rule in reduced normal form ([rnf](#_bookmark7)). Let Θ(*r*) d=ef *{φ ,...,φ }* be the

1 *s*

set of all disjuncts of *r*. For every *φj ∈* Θ(*r*), let

*θ*(*φj*)

d=ef *{p | t*(*i, j,* 0) = 1*}* and *θ* (*φ* )

d=ef *{p | t*(*i, j,* 1) = 1*}.*

That is, *θ*(*φj*) denotes the set of variables of *r* with positive occurrences in *φj*, and

*i j*

*i*

*θ*(*φj*) is the set of variables *pi* of *r* with the positive occurrence of *pi* in *φj*.

Historically the first algorithm for recognising admissible rules of S4 was based on the next theorem. Its formulation requires the following definition, which is taken from [[15](#_bookmark84)]. For every subset of disjuncts *W ⊆* Θ(rnf(*r*)), let *M*(rnf(*r*)*,W* ) denote the Kripke model in which *WM* =def *W* ,

*RM* d=ef

*{*(*φ ,φ* ) *| θ* (*φ* ) *⊆ θ* (*φ* )*}* and (*p ,φ* ) *∈ νM* def *p ∈ θ*(*φ* )*.*

1 2 2 1 *i j i j*

*⇐⇒*

**Theorem 3.4 (Theorem 3.9.6 [**[**18**](#_bookmark87)**])** *A rule* rnf(*r*) *is admissible for* S4 *iff for any set W ⊆* Θ(rnf(*r*))*, the model M*(rnf(*r*)*,W* ) *fails to have at least one of the following properties.*

*(a1) There is φj ∈ W such that M*(rnf(*r*)*,W* )*, φj |*= *p*0*. (a2) M*(rnf(*r*)*,W* )*, φj |*= *φj for all φj ∈ W.*

*(a3) For any subset D of M there exists φj ∈ W such that*

*θ*(*φj*) = *θ*(*φj*) *∪* *θ*(*φ*)*.*

*φ∈D*

Note that in [(a3)](#_bookmark11), *D* can be empty.

Theorem [3.4](#_bookmark9) implies that we can employ this algorithm for recognising the ad- missibility of a rule *r* in S4 [[15](#_bookmark84)].

Step 1. Reduce rule *r* to depth-one-form do(*r*).

Step 2. Construct from the do(*r*) the reduced form rnf(*r*).

Step 3. For every subset *W* of the set of disjuncts of rnf(*r*), check whether the conditions of Theorem [3.4](#_bookmark9) hold.

Step 4. If the conditions [(a1)](#_bookmark10)–[(a3)](#_bookmark11) of Theorem [3.4](#_bookmark9) are satisfied for some *W* , then *r*

is not admissible for S4, otherwise *r* is admissible for S4.

[Step 1](#_bookmark12) can be done in polynomial time, [Step 2](#_bookmark13) can be done in exponential time, and [Step 3](#_bookmark14) can be done in exponential time. This means the time complexity of the algorithm is bounded by a doubly-exponential function in the length of *r*. A more detailed complexity analysis based on the techniques from [[8](#_bookmark79),[9](#_bookmark77)] implies that the rule admissibility problem for S4 is coNExpTime-complete [[10](#_bookmark78)].

# Semantic Characterisation of Admissibility

We now give a semantic characterisation of admissibility in S4. We say that an S4u-model satisfies the *formula deﬁnable co-cover property* if the following (infinite) set of axioms hold (for *n >* 0):

(FCCP)

*∃x∀p* (*ν*( *p, x*) *→ ν*(*p, x*))

*∀x*1 *··· ∀xn∃x∀p* *R*(*x, x*1) *∧· · · ∧ R*(*x, xn*)*∧*

*ν*( *p, x*) *→* (*ν*(*p, x*) *∨ ν*( *p, x*1) *∨· · · ∨ ν*( *p, xn*)) *.*

Let *SFCCP* be the semantic specification consisting of the ([FCCP](#_bookmark15)) formulae and the formulae in *S*S4u . Let *FCCP*(Σ) be the class of all S4u-models of the signature Σ that satisfy all instances of the formulae of *SFCCP* under substitutions of *L*u -

S4

formulae in Σ for propositional variables.

Let S4u,+ be the modal logic with the language *L*u

S4

that has *SFCCP* as semantic

specification. The following theorem is a direct consequence of the definitions above and the fact that all *n*-characterising models Ch*n*(S4) for S4 satisfy the ([FCCP](#_bookmark15)) formulae.

**Theorem 4.1** S4*u,+ is a conservative extension of* S4*.*

Now we prove that S4u,+ has the effective finite model property. Let Σ be a *ﬁxed* signature and *M* an S4u,+-model of the signature Σ. We define the *ﬁltrated* (through Σ) model *M* as follows. The equivalence *∼* on the set *WM* is defined by

*w ∼ v ⇐⇒*

def

*τ* (*w*) = *τ* (*v*). Further, [*w*] =def *{v ∈ WM | w ∼ v}* and *WM* =def *{*[*w*] *| w ∈*

*WM }*. Next, *RM* =def *{*([*w*]*,* [*v*]) *| M, v |*= *ψ* implies *M, w |*= *ψ* for every *ψ ∈* Σ*}* and, finally, *νM* =def *{*(*ψ,* [*w*]) *|* (*ψ, w*) *∈ νM }* for every *ψ ∈* Σ.

The following lemma can be proved by induction on the structure of formulae in Σ by verifying that all semantic conditions in *SFCCP* hold.

**Lemma 4.2 (Filtration Lemma)** *M is an* S4*u,+-model of the signature* Σ*.* Note that by definition *M* is Σ-differentiated and it is finite whenever Σ is finite.

**Theorem 4.3 (The Effective Finite Model Property)** *Let φ be a formula and n the length of φ (i.e., the number of symbols in φ). If φ is satisﬁable in an* S4*u,+-model (of the signature* sub(*φ*)*) then it is satisﬁable in a ﬁnite* S4*u,+-model (of the signature* sub(*φ*)*) and its size does not exceed* 2*n.*

**Theorem 4.4** *α/β ∈* Adm(S4) *iff* [u]*α ∧ ¬β is unsatisﬁable in* S4*u,+.*

**Proof** Suppose *α/β* is not admissible and *p*1*,..., pn* are all the propositional vari- ables occurring in *α* and *β*. Then there is a model *M* —a definable variant of Ch*n*(S4) such that *M |*= *α/β*. This model has the co-cover property and it is routine to transform *M* into an S4u,+-model satisfying [u]*α ∧ ¬β*.

For the converse, let Σ =def sub([u]*α∧¬β*) and suppose [u]*α∧¬β* is satisfiable in an

S4u,+-model. Then it is satisfiable in a finite Σ-differentiated S4u,+-model *M* (of the

signature Σ), by the Filtration Lemma. Let sub (*α, β*) =def *{ γ | γ ∈* sub(*α, β*)*}∪*

sub(*α, β*) for any *L*S4

-formulae *α* and *β*. For every *w ∈ WM* let *τ*

(*w*) =def

*{γ ∈*

sub(*α, β*) *| M, w |*= *γ}* and

*γ*

*φ*(*w*) =

def

*γ∈*sub(*α,β*)

*qχ*(*γ*) *∧*

*γ∈*sub(*α,β*)

*γ*

*qχ*( *γ*)*,*

where *χ* is the characteristic function of the set *τ*(*w*). Let us consider the model

*M∗* d=ef

*⇐⇒*

*⟨WM∗ , RM∗ , νM∗ ⟩*, where *WM∗*

=def

*{φ*(*w*) *| w ∈ WM }*, (*φ*(*u*)*, φ*(*v*)) *∈*

*RM∗*

def

(*u, v*) *∈ RM* , and (*qγ*

*, φ*(*w*)) *∈ νM∗*

def

*M, w |*= *γ*. Each *φ*(*w*) is a

disjunct in reduced normal form rnf(*r*). Therefore we have that

*⇐⇒*

* *WM∗ ⊆* Θ(rnf(*r*)),
* *⟨WM∗ , RM∗ ⟩* is isomorphic to the underlying frame of *M* ,
* *M∗* satisfies conditions [(a1)](#_bookmark10)–[(a3)](#_bookmark11) of Theorem [3.4](#_bookmark9).

By Theorem [3.4](#_bookmark9), rnf(*r*) is not admissible, and hence neither is *r*.

# Synthesising a Tableau Calculus

We now apply the method of [[22](#_bookmark91)] to generate a sound and constructively complete tableau calculus for S4u,+. In order to apply the method we must ensure that the se- mantic specification *SFCCP* of S4u,+ is well-defined in the sense of [[22](#_bookmark91)]. First, *SFCCP* must be normalised in the following sense: (i) all the formulae specifying semantics are *L*u -open, i.e., they do not contain quantifiers of propositional variables, and

S4

(ii) all the formulae in *SFCCP* are divided into three groups: positive and negat- ive definitions of the semantics of the connectives of the logic, and a background theory that imposes frame conditions. It is also required that all formulae in the background theory do not contain any non-atomic modal terms.

Every definition of the *L*u connectives in *SFCCP* (in Figure [1](#_bookmark2) and ([1](#_bookmark4))) can be

S4

split into two implications. The resulting set of formulae can be divided into the required two groups of positive and negative connective definitions. The third group, the background theory of S4u,+, consists of formulae specifying the reflexivity and

transitivity for the relation *R* and the ([FCCP](#_bookmark15)) formulae.

The main difficulties for the normalisation of the specification *SFCCP* are the oc- currences of the non-atomic modal term *p* and the quantification of the variable *p* in ([FCCP](#_bookmark15)). To solve this problem we first replace every formula *ν*( *p, y*) by its semantic equivalent *∃z*(*R*(*y, z*) *∧ ν*(*p, z*)) and transform the resulting formulae into the prenex normal form. This gives us:

*∃x∀p∀y* (*R*(*x, y*) *∧ ν*(*p, y*)) *→ ν*(*p, x*)

*∀x*1 *··· ∀xn∃x∀p∀y∃z* *R*(*x, x*1) *∧· · · ∧ R*(*x, xn*)*∧*

(*R*(*x, y*) *∧ ν*(*p, y*)) *→* *ν*(*p, x*) *∨* (*ν*(*p, z*) *∧* (*R*(*x*1*, z*) *∨· · · ∨ R*(*xn, z*))) *.*

These formulae are still not *L*u -open formulae as required in [[22](#_bookmark91)] because of the

S4

*ν*(*¬p, x*)

*¬ν*(*p, x*)

*ν*(*p ∨ q, x*)

*ν*(*p, x*) *| ν*(*q, x*) *ν*( *p, x*)

*R*(*x, f* (*p, x*))*, ν*(*p, f* (*p, x*)) *ν*(*⟨*u*⟩p, x*)

*ν*(*p, f*u(*p*))

*¬ν*(*¬p, x*)

*ν*(*p, x*)

*¬ν*(*p ∨ q, x*)

*¬ν*(*p, x*)*, ¬ν*(*q, x*) *ν*(*¬ p, x*)

*¬R*(*x, y*) *| ¬ν*(*p, y*) *ν*(*¬⟨*u*⟩p, x*)*, y ≈ y*

*¬ν*(*p, y*)

Theory tableau rules:

*x ≈ x R*(*x, x*)

*x ≈ x, y ≈ y, z ≈ z*

*¬R*(*x, y*) *| ¬R*(*y, z*) *| R*(*x, z*)

Infinite set of ([FCCP](#_bookmark15)) tableau rules (*n >* 0):

(cc0): *p ≈ p, y ≈ y*

(cc*n*): *x*1 *≈ x*1*, ..., xn ≈ xn, y ≈ y*

*¬R*(*g*0*, y*) *| ¬ν*(*p, y*) *| ν*(*p, g*0)

0 *R*(*gn*(*x*)*, x*1)*, ..., R*(*gn*(*x*)*, xn*)

(cc*n*): *p ≈ p, x*1 *≈ x*1*, ... , xn ≈ xn, y ≈ y*

1 *¬R*(*gn*(*x*)*, y*) *| ¬ν*(*p, y*) *| ν*(*p, gn*(*x*)) *| R*(*x*1*, hn*(*p, x, y*))*, ν*(*p, hn*(*p, x, y*)) *| ···*

Closure tableau rules:

*ν*1(*p, x*)*, ¬ν*1(*p, x*)

*⊥*

*R*(*x, y*)*, ¬R*(*x, y*)

*⊥*

Figure 2. Generated tableau rules for S4u,+

quantifiers on *p*. However, using Skolemisation it is possible to eliminate all exist- ential quantifiers preceding the *p*-quantifiers in the formulae and then we can omit the quantifiers of *p*. In addition, we split the long formulae in two parts. We get:

(FCCP*'*)

*∀y* (*R*(*g*0*, y*) *∧ ν*(*p, y*)) *→ ν*(*p, g*0)

*∀x*1 *··· ∀xn* *R*(*gn*(*x*1*,..., xn*)*, x*1) *∧· · · ∧ R*(*gn*(*x*1*,..., xn*)*, xn*)

*∀x*1 *··· ∀xn∀y∃z* (*R*(*gn*(*x*1*,..., xn*)*, y*) *∧ ν*(*p, y*)) *→*

*ν*(*p, gn*(*x*1*,..., xn*)) *∨* (*ν*(*p, z*) *∧* (*R*(*x*1*, z*) *∨· · · ∨ R*(*xn, z*))) *.*

Here, *gn* (*n ≥* 0) denote the introduced Skolem symbols.

We use the notation *S'* for the semantic specification obtained from *SFCCP*

*FCCP*

where all ([FCCP](#_bookmark15)) formulae have been replaced by the corresponding ([FCCP*'*](#_bookmark18)) for- mulae. It is clear that for every first-order structure *M* , the universal closure of

*SFCCP* and the universal closure of *S'* are equisatisfiable in *M* . Hence, the

*FCCP*

transformed semantics *S'* is equivalent to *SFCCP* and axiomatises the same

*FCCP*

class of models.

Now we are ready to synthesise tableau calculi from *S'*

*FCCP*

. The generated

tableau rules are given in Figure [2](#_bookmark19). The symbols *f*, *f*u, *gn*, *hn* denote Skolem functions and *g*0 denotes a Skolem constant.

Let *T* be the tableau calculus consisting of the rules of Figure [2](#_bookmark19) and the standard tableau rules for equality listed in Figure [3](#_bookmark20). The equality tableau rules are obtained from the equality congruence axioms, which are always included in the background theory of any semantic specification, see [[22](#_bookmark91)].

*R*(*x, y*)

*x ≈ x, y ≈ y*

*¬R*(*x, y*) *x ≈ x, y ≈ y*

*¬ν*(*p, x*)

*p ≈ p, x ≈ x*

*ν*(*p, x*)

*p ≈ p, x ≈ x*

*R*(*x, y*)*, x ≈ z*

*R*(*z, y*)

*R*(*x, y*)*, y ≈ z*

*R*(*x, z*)

*ν*(*p, x*)*, x ≈ y*

*ν*(*p, y*)

*x ≈ y y ≈ x*

*x ≈ y, y ≈ z x ≈ z*

Figure 3. Equality congruence rules for predicates occurring in *S*S4u .

Given a formula *φ*, and assuming our aim is to determine the satisfiability of *φ*, the start of any tableau derivation is the formula *ν*(*φ, a*), where *a* is an arbitrary constant *a* that does not occur in the rules of *T* . *a* can be viewed as a Skolem constant introduced for *∃x* in the formula *∃x ν*(*φ, x*). The rules of the calculus are applied top-down in the familiar way.

We assume the following definitions from [[21](#_bookmark90),[22](#_bookmark91)]. Let *T* denote a tableau calculus and *φ* is a formula. We denote by *T* (*φ*) a finished tableau derivation for testing the satisfiability of *φ*. That is, all branches in the tableau derivation are fully expanded and all applicable rules of *T* have been applied in *T* (*φ*). As usual we assume that all the rules of the calculus are applied non-deterministically. A branch of a tableau

derivation is *closed* if a closure rule has been applied, otherwise the branch is called

*open*. The tableau derivation *T* (*φ*) is *closed* if all its branches are closed and *T* (*φ*) is *open* otherwise. A calculus *T* is *sound* (for a logic *L*) iff for any formula *φ*, each *T* (*φ*) is open whenever *φ* is satisfiable in an *L*-model. *T* is *complete* iff for any unsatisfiable formula *φ* there is a *T* (*φ*) which is closed. *T* is *constructively complete* (for *L*) iff for any open branch in a finished tableau derivation in *T* it is possible to construct an *L*-model from terms of the branch such that the model reflects all the formulae occurring in the branch. (Constructive completeness is a stronger notion than completeness.) *T* is said to be *terminating* if every finished open tableau derivation in *T* has a finite open branch.

It is easy to check that the specification *S'* for S4u,+ is well-defined in the

*FCCP*

sense of [[22](#_bookmark91)]. A consequence of [[22](#_bookmark91), Theorems 1 and 2] is this result.

**Theorem 5.1** *T is a sound and constructively complete calculus for* S4*u,+.*

# A Refined Tableau Calculus

The generated calculus *T* can be refined as follows. First, it is possible to refine the calculus by moving negated conclusions in certain rules up to premise positions. We move formulae that contain only propositional variables and do not contain any complex modal terms upwards. It is not difficult to check for each rule that the condition given in [[22](#_bookmark91), Theorem 3] for this refinement to preserve soundness and constructive completeness is true.

Second, we can apply the second refinement described in [[22](#_bookmark91), Section 5]. For this

we extend the language *L*u by introducing a countable set *{i, j, k,.. .}* of nominal

S4

variables and logical connectives (acting on nominals) which correspond to Skolem functions and constants. The @ operator can be introduced and specified in such a

way that @*iφ* =def [u](*i → φ*) for every nominal term *i* and formula *φ* (of the extended

(*∨*): @*i*(*p ∨ q*)

@*ip |* @*iq*

@*i p*

():

@*i f*(*p, i*)*,* @*f* (*p,i*)*p*

@*i⟨*u*⟩p*

(*⟨*u*⟩*):

@*f*u(*p*)*p*

(*¬*): @*i¬¬p*

@*ip*

@*i¬*(*p ∨ q*)

(*¬∨*):

@*i¬p,* @*i¬q*

(*¬* ): @*i¬ p,* @*i j*

@*j ¬p*

(*¬⟨*u*⟩*): @*i¬⟨*u*⟩p,* @*j j*

@*j ¬p*

Theory tableau rules:

@*ii*

(refl):

@*i i*

(trans): @*i j,* @*j k*

@*i k*

Infinite set of ([FCCP*'*](#_bookmark18)) tableau rules (*n >* 0):

(cc*'*0):

(cc*'*0): @*g*0 *i,* @*ip*

0 @*g*0 *g*0 1 @*g*0 *p*

(cc*'n*): @*i*1 *i*1*, ... ,* @*in in*

0

@*gn* (*i*)

*i*1*, ...,* @

*gn* (*i*)

*in*

(cc*'n*): @*gn* (*i*) *j,* @*j p*

1

@*gn* (*i*)

*p |* @*i*1 *hn*(*p, i, j*)*,* @

*hn* (*p,i,j*)

*p | ··· |* @*in*

*hn*(*p, i, j*)*,* @

*hn* (*p,i,j*)*p*

Closure tableau rules:

(*⊥*): @*ip,* @*i¬p*

*⊥*

Figure 4. Refined tableau rules for S4u,+

(refl*≈*): @*ip*

@*ii*

(sym*≈*): @*ij*

@*j i*

(trans*≈*): @*ij,* @*jk*

@*ik*

(con*≈* ): @*ip,* @*ij*

0

@*j p*

(con*≈* ): @*i j,* @*jk*

1 @ *k*

*i*

Figure 5. Refined equality congruence rules

language). It is not difficult to see that the semantics of all the connectives of the extended language can be represented in the language itself (see [[22](#_bookmark91), Section 5]).

Summing up, the refined rules we obtain are given in Figure [4](#_bookmark22). In these rules, *i, j, k, i*1*,..., in* denote nominal variables, and *f*, *f*u, *gn*, and *hn* (*n ≥* 0) denote ‘nominal functions’ which correspond to the Skolem functions with same names.

Let *T* + be the tableau calculus which consists of the rules of Figure [4](#_bookmark22) and the refined equality rules given in Figure [5](#_bookmark23). In *T* +, a tableau derivation for testing the satisfiability of *φ* starts with a formula @*i*0 *φ* where *i*0 is a *fresh* nominal constant.

**Theorem 6.1** *T* + *is a sound and constructively complete tableau calculus for* S4*u,+.*

# A Terminating Tableau Calculus

Our proof of Theorem [4.3](#_bookmark16) that S4u,+ has the effective finite model property uses a filtration argument. That is, in the terminology of [[21](#_bookmark90)], S4u,+ admits finite filtration. Using the results of [[21](#_bookmark90)], this means that the tableau calculi generated in Section [5](#_bookmark18) and [6](#_bookmark21) can be turned into terminating tableau calculi. In particular, we are interested only in the refined calculus *T* +.

Adding the following unrestricted blocking rule to *T* + gives a terminating tableau calculus.

@*ii,* @*jj*

(ub): @ *j |* @ *¬j*

*i* *i*

The conditions that blocking must satisfy are:

(b1) The rules () and ([*⟨*u*⟩*](#_bookmark21)) are never applied to formulae of the form @*i j* and, respectively, @*i⟨*u*⟩j* where *j* is a nominal term.

(b2) If @*ij* appears in a branch and *i < j* (i.e., nominal *i* appeared strictly earlier than nominal *j* in the derivation) then all further applications of the tableau

rules which produce new nominals (in our case the (), ([*⟨*u*⟩*](#_bookmark21)), ([cc*'*0](#_bookmark21)) and ([cc*'n*](#_bookmark21))

[0](#_bookmark21) [0](#_bookmark21)

rules) to the formulae with occurrences of *j* are not performed within the branch.

(b3) In every open branch there is some node from which point onwards before any application of any tableau rule that produces new nominals (i.e., the (),

([*⟨*u*⟩*](#_bookmark21)), ([cc*'*0](#_bookmark21)) and ([cc*'n*](#_bookmark21)) rules) all possible applications of the ([ub](#_bookmark24)) rule have

[0](#_bookmark21) [0](#_bookmark21)

been performed.

We denote the extended calculus by *T*S4u,+ .

Since *T* + is sound and constructively complete for S4u,+, and S4u,+ admits finite

filtration the results in [[20](#_bookmark89)] allow us to state:

**Theorem 7.1** *T*S4*u,+ is a sound, (constructively) complete and terminating tableau calculus for* S4*u,+.*

Let *T*S4u be the tableau calculus which consists of the same set of rules as *T*S4u,+

but excludes the ([FCCP*'*](#_bookmark18)) rules: ([cc*'*0](#_bookmark21)), ([cc*'*0](#_bookmark21)), ([cc*'n*](#_bookmark21)), and ([cc*'n*](#_bookmark21)). Applying tableau

[0](#_bookmark21) [1](#_bookmark21) [0](#_bookmark21) [1](#_bookmark21)

synthesis to S4u in a similar way gives the following result.

**Theorem 7.2** *T*S4*u is a sound, (constructively) complete and terminating tableau* *calculus for* S4*u.*

# A Tableau Method for Testing Rule Admissibility

Putting all the results together (in particular Theorems [3.1](#_bookmark6), [4.4](#_bookmark17), [7.1](#_bookmark25) and [7.2](#_bookmark26)) here is a method for determining whether a modal rule is admissible in S4, or not.

Step 1. Given an S4-rule *α/β*, rewrite it to [u]*α ∧ ¬β*.

Step 2. Use the tableau calculus *T*S4u to test the satisfiability of [u]*α ∧ ¬β* in S4u.

Step 3. If *T*S4u ([u]*α ∧ ¬β*) is closed, i.e., [u]*α ∧ ¬β* is unsatisfiable in S4u, then stop and return **‘derivable’**.

1. @*i ¬⟨*u*⟩*(*¬ p ∨ ¬ ¬p*) given
2. @*i i*4 (),[18](#_bookmark55):

def

4 2

0 2 *i*

= *f* (*¬p, i* )

1. @*i*0 *i*0 ([refl*≈*](#_bookmark22)),[1](#_bookmark29)
2. @*i*0 *¬*(*¬ p ∨ ¬ ¬p*) ([*¬⟨*u*⟩*](#_bookmark21)),[1](#_bookmark29),[2](#_bookmark30)
3. @*i*0 *¬¬ p* ([*¬∨*](#_bookmark21)),[3](#_bookmark31)
4. @*i*0 *¬¬ ¬p* ([*¬∨*](#_bookmark21)),[3](#_bookmark31)
5. @*i*0 *p* ([*¬*](#_bookmark21)),[4](#_bookmark32)
6. @*i*0 *¬p* ([*¬*](#_bookmark21)),[5](#_bookmark33)
7. @*i*4 *¬p* (),[18](#_bookmark55)
8. @*i*4 *i*4 ([refl*≈*](#_bookmark22)),[32](#_bookmark30)
9. )@*i*0 *i*3 ([ub](#_bookmark24)),[2](#_bookmark30),[30](#_bookmark67)
10. @*i*3 *i*0 ([sym*≈*](#_bookmark22)),[34](#_bookmark32)
11. @*i*2 *i*0 ([con*≈*1](#_bookmark22)),[28](#_bookmark65),[35](#_bookmark33)
12. @*i*2 *i*2 ([trans](#_bookmark21)),[11](#_bookmark39),[36](#_bookmark34)
13. @*i i*1 (),[6](#_bookmark34):

def

1 0

1. )@*i*2 *i*4 ([ub](#_bookmark24)),[13](#_bookmark44),[33](#_bookmark31)

0 *i* = *f* (*p, i* )

... *...* ..................................

1. @*i*1 *p* (),[6](#_bookmark34)
2. @*i*1 *i*1 ([refl*≈*](#_bookmark22)),[9](#_bookmark35)
3. @*i i*2 (),[7](#_bookmark36): *i*2 def *f*(*¬p, i*0)

0 =

1. @*i*2 *¬p* (),[7](#_bookmark36)
2. @*i*2 *i*2 ([refl*≈*](#_bookmark22)),[12](#_bookmark42)
3. @*i*2 *¬*(*¬ p ∨ ¬ ¬p*) ([*¬⟨*u*⟩*](#_bookmark21)),[1](#_bookmark29),[13](#_bookmark44)
4. @*i*2 *¬¬ p* ([*¬∨*](#_bookmark21)),[14](#_bookmark46)
5. @*i*2 *¬¬ ¬p* ([*¬∨*](#_bookmark21)),[14](#_bookmark46)
6. @*i*2 *p* ([*¬*](#_bookmark21)),[15](#_bookmark48)
7. @*i*2 *¬p* ([*¬*](#_bookmark21)),[16](#_bookmark50)

Satisfiable in S4u ..................... The rule is **not derivable** .............

1. @*g*0 *g*0 ([cc*'*0](#_bookmark21))

[0](#_bookmark21)

1. @*g*0 *¬*(*¬ p ∨ ¬ ¬p*) ([*¬⟨*u*⟩*](#_bookmark21)),[1](#_bookmark29),[35](#_bookmark33)
2. @*g*0 *¬¬ p* ([*¬∨*](#_bookmark21)),[40](#_bookmark40)
3. @*g*0 *¬¬ ¬p* ([*¬∨*](#_bookmark21)),[40](#_bookmark40)
4. @*g*0 *p* ([*¬*](#_bookmark21)),[41](#_bookmark43)
5. @*g*0 *¬p* ([*¬*](#_bookmark21)),[42](#_bookmark45)
6. @*g i*5 ....... (),[43](#_bookmark47): 5 def 0

0 *i* = *f* (*p, g* )

1. @*i*5 *p* (),[43](#_bookmark47)
2. )@*i*0 *i*1 ([ub](#_bookmark24)),[2](#_bookmark30),[10](#_bookmark37)
3. @*g i*6 (),[44](#_bookmark49):

def

6 0

0 *i* = *f* (*¬p, g* )

1. @*i*1 *i*0 ([sym*≈*](#_bookmark22)),[19](#_bookmark54)
2. @*i*0 *p* ([con*≈*0](#_bookmark22)),[9](#_bookmark35),[20](#_bookmark57)
3. @*i*1 *i*1 ([con*≈*0](#_bookmark22)),[8](#_bookmark38),[19](#_bookmark54)
4. @*i*6 *¬p* (),[44](#_bookmark49)
5. @*g*0 *p* ([cc*'*0](#_bookmark21)),[45](#_bookmark51),[46](#_bookmark53)

[1](#_bookmark21)

1. @*g*0 *¬p* ([cc*'*0](#_bookmark21)),[47](#_bookmark56),[48](#_bookmark58)
2. @*i*0 *i*0 ([con*≈*1](#_bookmark22)),[8](#_bookmark38),[20](#_bookmark57)

Unsatisfiable in S4

u,+

[1](#_bookmark21)

. [49](#_bookmark60),[50](#_bookmark61)

1. )@*i*0 *i*2 ([ub](#_bookmark24)),[2](#_bookmark30),[13](#_bookmark44)
2. @*i*2 *i*0 ([sym*≈*](#_bookmark22)),[24](#_bookmark62)
3. @*i*0 *¬p* ([con*≈*0](#_bookmark22)),[12](#_bookmark42),[25](#_bookmark63)

Unsatisfiable [21](#_bookmark59),[26](#_bookmark64)

1. )@*i*0 *¬i*2 ([ub](#_bookmark24)),[2](#_bookmark30),[13](#_bookmark44)
2. )@*i*2 *¬i*4 ([ub](#_bookmark24)),[13](#_bookmark44),[33](#_bookmark31)

. Similarly to [39](#_bookmark41)–[50](#_bookmark61)

Unsatisfiable in S4u,+ .................

1. )@*i*0 *¬i*3 ([ub](#_bookmark24)),[2](#_bookmark30),[30](#_bookmark67)

. . . Similarly to [39](#_bookmark41)–[50](#_bookmark61)

1. @*i*2

*i*3

. (),[17](#_bookmark52):

def

*i*3 = *f*(*p, i*2)

Unsatisfiable in S4u,+ ..................

1. )@*i ¬i*1 ([ub](#_bookmark24)),[2](#_bookmark30),[10](#_bookmark37)
2. @*i*3 *p* (),[17](#_bookmark52)
3. @*i*3 *i*3 ([refl*≈*](#_bookmark22)),[29](#_bookmark66)

0

. . . Similarly to [39](#_bookmark41)–[50](#_bookmark61)

Unsatisfiable in S4u,+ ...................... The rule is **not derivable and admissible**

Figure 6. A derivation for testing admissibility of *p ∧ ¬p/⊥*.

Step 4. Otherwise (i.e., *T*S4u ([u]*α ∧ ¬β*) is open), *continue* the tableau derivation with the rules in *T*S4u plus the rules ([cc*'*0](#_bookmark21)), ([cc*'*0](#_bookmark21)), ([cc*'n*](#_bookmark21)), and ([cc*'n*](#_bookmark21)). In

[0](#_bookmark21) [1](#_bookmark21) [0](#_bookmark21) [1](#_bookmark21)

particular, continue the derivation with a finite open branch of *T*S4u ([u]*α ∧*

*¬β*) using the rules of *T*S4u,+ until the derivation stops.

Step 5. If *T*S4u,+ ([u]*α ∧ ¬β*) is closed then return **‘not derivable and admissible’**.

Otherwise, i.e., if *T*S4u,+ ([u]*α ∧ ¬β*) is open, return **‘not admissible’**.

The answers returned by the method are either **‘derivable’**, **‘not derivable and admissible’**, or **‘not admissible’**. If a rule is derivable it is also admissible but not conversely.

Figure [6](#_bookmark68) demonstrates the algorithm for the rule *p ∧ ¬p/⊥*. The [Step 1](#_bookmark28) rewrites the rule to *¬⟨*u*⟩*(*¬ p ∨ ¬ ¬p*) modulo standard modal equivalences. The black triangles in the figure denote branching points in the derivation. A branch expansion after a branching point is indicated by appropriate indentation.

# Concluding Remarks

A major difficulty in dealing with S4-admissibility, is that the theory of S4-ad- missible rules does not have the finite approximation property [[11](#_bookmark80)] in this sense:

for a rule *r ∈/* Adm(S4), there is no single finite Kripke model *M separating r*

from Adm(S4), (i.e., such that *M |*= Adm(S4), but *M |*= *r*). Therefore the tableau

algorithm that we have introduced in this paper builds open branches that represent a (possibly) infinite Adm(S4)-model and is a counter-model to a non-admissible rule. The subsequent filtration provides a finite model (not necessarily an Adm(S4)-

model) witnessing the refutation.

A known algorithm that can handle S4-admissibility appears in Zucchelli [[24](#_bookmark93)] and is based on the research of Ghilardi [[3](#_bookmark72),[4](#_bookmark73),[5](#_bookmark75),[6](#_bookmark76)] on projective approximations. This algorithm is based on the algorithm from [[5](#_bookmark75)] for IPC, which combines resolution and tableau approaches for finding projective approximations of a formula and relies on the existence of an algorithm for theorem proving. This algorithm (as well as its precursor for IPC [[5](#_bookmark75)]) is specifically constructed for describing general solutions (maximal general unifiers, maximal since there can be more than one) for matching and unification problems (all other solutions can be obtained as substitution variants of general solutions). Applicability of this algorithm to the admissibility problem is a side effect, depending on some specific properties of S4 (see [[6](#_bookmark76)]). In contrast, the original algorithm of [[15](#_bookmark84)] can be used to find only *some* solution of matching and unification problems in S4. In particular, this can be done through the relation:

an equation *α ≡ β* is solvable iff a rule *α ≡ β/⊥* is not admissible.

We expect that our method can be modified for finding the general solutions of logical equations as well.

Recently Wolter and Zakharyaschev [[23](#_bookmark92)] showed that modal logics with the *universal modality* that are situated between Ku and K4u are undecidable with respect to admissibility and even with respect to only unification. They posed the

question [[23](#_bookmark92)] whether the logic S4u is decidable with respect to admissibility. This

question is solved positively in Rybakov [[19](#_bookmark88)]. We strongly believe that our tableau method can also be extended to rule admissibility in S4u.

In fact, our method of replacing the first-order co-cover condition with its *formu- laic* variant (which is also first-order but in the extended language) is not restricted to S4. We expect that it can be modified to deal with a number of other modal logics, especially those covered by the general theory of [[18](#_bookmark87)]. Similarly, it can be ap- plied to their superintuitionistic counterparts (either through Go¨del’s translation or directly) and to transitive modal logics augmented with the universal modality [[19](#_bookmark88)].

# References

1. S. Babenyshev. The decidability of admissibility problems for modal logics S4.2 and S4.2Grz and superintuitionistic logic KC. *Algebra Logic*, 31(4):205–216, 1992.
2. H. Friedman. One hundred and two problems in mathematical logic. *J. Symb. Log.*, 40(3):113–130, 1975.
3. S. Ghilardi. Unification in intuitionistic logic. *J. Symb. Log.*, 64(2):859–880, 1999.
4. S. Ghilardi. Best solving modal equations. *Ann. Pure Appl. Logic*, 102(3):183–198, 2000.
5. S. Ghilardi. A resolution/tableaux algorithm for projective approximations in IPC. *Log. J. IGPL*, 10(3):229–243, 2002.
6. S. Ghilardi and L. Sacchetti. Filtering unification and most general unifiers in modal logic. *J. Symb. Log.*, 69(3):879–906, 2004.
7. R. Harrop. Concerning formulas of the types *a → b ∨ c*, *a → ∃xb*(*x*) in intuitionistic formal system.

*J. Symb. Log.*, 25:27–32, 1960.

1. R. Iemhoff. On the admissible rules of intuitionistic propositional logic. *J. Symb. Log.*, 66(1):281–294, 2001.
2. E. Jeˇr´abek. Admissible rules of modal logics. *J. Log. Comput.*, 15(4):411–431, 2005.
3. E. Jeˇr´abek. Complexity of admissible rules. *Arch. Math. Logic*, 46(2):73–92, 2007.
4. V. R. Kiyatkin, V. V. Rybakov, and T. Oner. On finite model property for admissible rules.

*Mathematical Logic Quarterly*, 45:505–520, 1999.

1. P. Lorenzen. *Einfu¨hrung in die operative Logik und Mathematik*. Springer, 1955.
2. G. Mints. Derivability of admissible rules. *Journal of Soviet Mathematics*, 6(4):417–421, 1976.
3. P. Roziere. Admissible and derivable rules. *Mathematical Structures in Computer Science*, (3):129–136, 1993.
4. V. V. Rybakov. A criterion for admissibility of rules in modal system S4 and the intuitionistic logic.

*Algebra Logic*, 23(5):369–384, 1984.

1. V. V. Rybakov. Semantic admissibility criteria for deduction rules in S4 and Int. *Mat. Zametki*, 50(1):84–91, 1991.
2. V. V. Rybakov. Rules of inference with parameters for intuitionistic logic. *J. Symb. Log.*, 57(3):912–923, 1992.
3. V. V. Rybakov. *Admissibility of logical inference rules*, vol. 136 of *Studies in Logic and the Foundations* *of Mathematics*. Elsevier, 1997.
4. V. V. Rybakov. Logics with universal modality and admissible consecutions. *J. Appl. Non-Classical* *Log.*, 17(3):381–394, 2007.
5. R. A. Schmidt and D. Tishkovsky. Using tableau to decide expressive description logics with role negation. In *ISWC07*, vol. 4825 of *Lect. Notes Comput. Sci.*, pp. 438–451. Springer, 2007.
6. R. A. Schmidt and D. Tishkovsky. A general tableau method for deciding description logics, modal logics and related first-order fragments. In *IJCAR08*, vol. 5195 of *Lect. Notes Comput. Sci.*, pp. 194–

209. Springer, 2008.

1. R. A. Schmidt and D. Tishkovsky. Automated synthesis of tableau calculi. In *TABLEAUX09*, vol. 5607 of *Lect. Notes Artif. Intell.*, pp. 310–324. Springer, 2009.
2. F. Wolter and M. Zakharyaschev. Undecidability of the unification and admissibility problems for modal and description logics. *ACM Transdactions on Computational Logic*, 9(4):1–20, 2008.
3. D. Zucchelli. Studio e realizzazione di algoritmi per l’unificazione nelle logiche modali. Laurea specialistica in informatica (Masters Thesis), Universit`a degli Studi di Milano, 2004. In Italian. Supervisor: Silvio Ghilardi.