Tableau-Based Model Checking in the Propositional Mu-Calculus*

Rance Cleaveland
Department of Computer Science
Box 8206
North Carolina State University
Raleigh, North Carolina 27695-8206, USA

Abstract

This paper describes a procedure, based around the construction of tableau proofs, for determining whether finite-state systems enjoy properties formulated in the propositional mu-calculus. It presents a tableau-based proof system for the logic and proves it sound and complete, and it discusses techniques for the efficient construction of proofs that states enjoy properties expressed in the logic. The approach is the basis of an ongoing implementation of a model checker in the Concurrency Workbench, an automated tool for the analysis of concurrent systems.

1 Introduction

One area of program verification that has proven amenable to automation involves the analysis of finite-state processes. While computer systems in general are not finite-state, many interesting ones, including a variety of communication protocols and hardware systems, are, and their finitary nature enables the development and implementation of decision procedures that test for various properties.

Model checking has proven a useful means for automatically ascertaining the correctness of such systems [2, 7, 11, 13, 27]. In this approach, one uses a logic to specify the desired properties of a system and a decision procedure to determine automatically if the "start state" of the system in question satisfies these formulas. Various temporal and modal logics have been investigated, and several case studies have pointed to the practical benefits of this form of verification [3, 4, 5].

One particularly expressive logic is the *propositional mu-calculus* [16]. A wide variety of "branching time" logics [12, 17, 21], including dynamic logic [14] and many temporal logics, have uniform encodings in this logic, and it also may be used to characterize fully the behavior of finite-state processes [22]; these facts make it a natural candidate for use in model checking. In this paper we develop a sound and complete proof system, and from it

*Research supported by British Science and Engineering Research Council grant GC/D69464. The results in this paper were obtained while the author was a research associate at the University of Sussex in Brighton, England.

$$
\begin{array}{rcl}
\Phi & ::= & A \\
& | & X \\
& | & \neg\Phi \\
& | & \Phi \vee \Phi \\
& | & \langle a \rangle \Phi \\
& | & \nu X.\Phi
\end{array}
$$

X may not appear negatively in Φ in the proposition $\nu X.\Phi$.

Figure 1: The syntax of the propositional mu-calculus.

a model checker, for determining whether states in a finite-state system satisfy propositions in the propositional mu-calculus, and we discuss techniques for the efficient construction of *tableau*, or *top-down*, proofs in this logic. Such a proof-based approach has several useful properties. In contrast to semantics-based strategies [13], our technique does not require that every state in the system be examined in order to determine if a particular state has a property, unless the property renders such an analysis necessary. Another is that features of proof systems that have been developed for similar logics may be carried over into our proof system, and therefore to the the model checker. Of particular interest in this regard are the *compositional* proof systems for sublogics of the mu-calculus developed by Stirling [23, 24], which allow properties of systems consisting of parallel components to be deduced on the basis of the properties enjoyed by each component. Typically, model checkers verify systems of parallel processes by modeling parallelism as interleaving; this can result in a combinatorial explosion in the size of the state space of the system as a function of the sizes of its components. Compositional model checking represents one approach to overcoming this problem [6]; however, developing such model checkers is very difficult. By applying Stirling's techniques to the proof system contained in this paper, it is likely that a compositional proof system for the full mu-calculus, and hence a compositional model checker, can be developed.

The remainder of the paper is organized as follows. In section 2 the syntax and semantics of the propositional mu-calculus are presented. Section 3 contains a presentation of the tableau system and a sample tableau proof, while section 4 establishes its soundness and completeness. Section 5 briefly considers techniques for an efficient implementation, and the last section contains our conclusions and directions for future work.

2 Syntax and Semantics

Syntactically, the propositional mu-calculus is parameterized with respect to a set $\mathcal{A}$ of *atomic formulas*, a set $\mathcal{V}$ (disjoint from $\mathcal{A}$) of *propositional variables*, and a set Act of *actions*. In what follows $\mathcal{A}$ will be ranged over by $A, ...$, Act will be ranged over by $a, ...$, $\mathcal{V}$ by $X, ...$, and formulas by $\Phi, ...$. Figure 1 describes the syntax of propositions. The symbols $\neg$ and $\vee$ represent negation and disjunction, respectively, while $\langle a \rangle$ is a *modal operator* indexed by action a. The formula $\nu X.\Phi$ is a recursive formula; the recursion operator ν binds all free occurrences of X in Φ, in the usual sense.[1] The syntactic restriction on the body of $\nu X.\Phi$ stipulates that any occurrences of X in Φ must occur inside the scope of an even number of negations.

We shall use the standard conventions for representing $\wedge$ and $\Rightarrow$. The proposition $[a]\Phi$ is derived notation for $\neg\langle a\rangle\neg\Phi$, and $\mu X.\Phi$ for $\neg\nu X.\neg\Phi[\neg X/X]$, where $\Phi[\Gamma/X]$ represents the simultaneous replacement of free occurrences of X in Φ by Γ, with bound variables in Φ renamed as necessary to prevent capture of free variables in Γ.

The following standard definitions are useful.

Definition 2.1 *Let Φ be a formula. The* length *of Φ, $|\Phi|$, is defined inductively as follows.*

1. $\Phi \in \mathcal{A} \cup \mathcal{V} \Rightarrow |\Phi| = 1$.
2. Φ *is* $\neg\Phi'$, $\langle a\rangle\Phi'$, $\mu X.\Phi' \Rightarrow |\Phi| = |\Phi'| + 1$.
3. Φ *is* $\Phi_1 \vee \Phi_2 \Rightarrow |\Phi| = |\Phi_1| + |\Phi_2| + 1$.

Definition 2.2 *Let Φ and Γ be formulas. The* immediate subterm relation *is defined by: $\Gamma \prec_I \Phi$ exactly when one of the following hold.*

1. Φ *is* $\neg\Gamma$.
2. Φ *is* $\Gamma \vee \Phi'$ *or* $\Phi' \vee \Gamma$ *for some* Φ'.
3. Φ *is* $\langle a\rangle\Gamma$.
4. Φ *is* $\nu X.\Gamma$.

The *strict subterm relation*, $\prec$, is defined as $\prec_I^+$, the transitive closure of $\prec_I$, while $\preceq$ is defined as $\prec_I^*$, the transitive and reflexive closure of $\prec_I$.

Formulas are given meaning relative to a *transition system* and a function interpreting atomic propositions. Transition systems may be thought of as representations of the operational behavior of processes; formally, they are triples of the form $\langle \mathcal{S}, Act, \rightarrow\rangle$, where $\mathcal{S}$ is a set of *states*, Act a set of *actions*, and $\rightarrow$ a relation, called the *transition relation*, on $\mathcal{S} \times Act \times \mathcal{S}$ representing the state transitions resulting from the "execution" of actions. We shall write $s \xrightarrow{a} s'$ in lieu of $\langle s, a, s'\rangle \in\rightarrow$, and we shall sometimes say that s *has an a-transition* if $s \xrightarrow{a} s'$ for some s'. Models for the mu-calculus are quadruples of the form $\langle \mathcal{S}, Act, \rightarrow, V\rangle$, where $\langle \mathcal{S}, Act, \rightarrow\rangle$ is a transition system and V is a function, called the *valuation*, mapping $\mathcal{A}$ to sets of states. We shall also use *environments*, which map variables to sets of states, as a means of interpreting free propositional variables. If e is an environment, then $e[X \mapsto S]$ represents the environment e with X "updated" to S.

Semantically, propositions correspond to the sets of states for which they are "true". The meaning function $[\![-]\!]_M$, where M is a model, is described in figure 2; in the figure, and in the remainder of the paper, we omit explicit reference to M when the model is clear from the context. Atomic formulas, variables, negation and disjunction are interpreted in the

[1] It should be noted that this account of the logic differs slightly from the standard account in that $\nu X.\Phi$ corresponds to a *greatest fixed point* operator, whereas the usual version of the logic has a least fixed point operator that is usually written $\mu X.\Phi$. These logics are, however, expressively equivalent.

$$
\begin{array}{rcl}
[\![A]\!]e & = & V(A) \\
[\![X]\!]e & = & e(X) \\
[\![\neg\Phi]\!]e & = & \mathcal{S} - [\![\Phi]\!]e \\
[\![\Phi_1 \vee \Phi_2]\!]e & = & [\![\Phi_1]\!]e \cup [\![\Phi_2]\!]e \\
[\![\langle a\rangle\Phi]\!]e & = & \pi_a([\![\Phi]\!]e), \text{ where } \pi_a(S) = \{\, s' \mid \exists s \in S.\ s' \xrightarrow{a} s \,\} \\
[\![\nu X.\Phi]\!]e & = & \bigcup\{\, S \subseteq \mathcal{S} \mid S \subseteq [\![\Phi]\!]e[X \mapsto S] \,\}
\end{array}
$$

Figure 2: The semantics of propositions.

obvious fashion, while the modal proposition $\langle a\rangle\Phi$ represents the set of states having an a-transition into a state contained in the meaning of Φ.

The interpretation of $\nu X.\Phi$ is somewhat more complicated. For any set $\mathcal{X}$, $\langle 2^{\mathcal{X}}, \subseteq, \cup, \cap\rangle$ forms a complete lattice. A function ϕ over this lattice is *monotonic* if, whenever $X_1 \subseteq X_2, \phi(X_1) \subseteq \phi(X_2)$. By the Tarski-Knaster theorem [26], any monotonic function ϕ over this lattice has a greatest fixed point, $\nu\phi$, and a least fixed point, $\mu\phi$, given by

$$
\begin{array}{rcl}
\nu\phi & = & \bigcup\{\, S \subseteq \mathcal{X} \mid S \subseteq \phi(S) \,\}, \text{ and} \\
\mu\phi & = & \bigcap\{\, S \subseteq \mathcal{X} \mid \phi(S) \subseteq S \,\}.
\end{array}
$$

As the next result shows, the syntactic restrictions on Φ guarantee that, given an environment e, the function ϕ defined by

$$\phi(S) = [\![\Phi]\!]e[X \mapsto S]$$

is monotonic over the lattice defined by $2^{\mathcal{S}}$ and hence has a greatest fixed point; this greatest fixed point is taken as the meaning of $\nu X.\Phi$.

Lemma 2.3 *Suppose X does not appear negatively in Φ. Then the function $\phi(S)$ defined by $[\![\Phi]\!]e[X \mapsto S]$ is monotonic.*

Proof. Define a function $\phi(S)$ over $2^{\mathcal{S}}$ to be *anti-monotonic* if $S_1 \subseteq S_2$ implies that $\phi(S_2) \subseteq \phi(S_1)$. The lemma is a consequence of the following slightly stronger result.

> Let $\phi(S) = [\![\Phi]\!]e[X \mapsto S]$. If X does not appear negatively in Φ then ϕ is monotonic, and if X does not appear positively in Φ then ϕ is anti-monotonic.

The proof follows from the monotonicity of $\vee$ and $\langle a\rangle$ and the anti-monotonicity of $\neg$ by a straightforward induction on the structure of Φ. $\square$

The next lemma establishes a connection between the (syntactic) notion of substitution and the (semantic) notion of function application. This leads to a corollary about the semantics of unrolling recursive propositions; from these results, it is possible to establish that $[\![\mu X.\Phi]\!]e$ corresponds to $\mu\phi$, where $\phi(S) = [\![\Phi]\!]e[X \mapsto S]$.

Lemma 2.4 *Let Φ and Γ be formulas, X be a variable, e and environment, and ϕ the function $\phi(S) = [\![\Phi]\!]e[X \mapsto S]$. Then $[\![\Phi[\Gamma/X]]\!]e = \phi([\![\Gamma]\!]e)$.*

Proof. The proof is by induction on the structure of Φ. The induction hypothesis states that if $\Phi' \prec \Phi$, then for all e, $[\![\Phi'[\Gamma/X]]\!]e = [\![\Phi']\!]e[X \mapsto [\![\Gamma]\!]e]$. Most cases are routine; here we consider the case when Φ is $\nu Y.\Phi'$. From the definition of substitution we may assume that Y is not free in Γ. Given the definition of $[\![-]\!]$ it follows that

$$\begin{aligned}
[\![\Phi[\Gamma/X]]\!]e &= \bigcup\{\, S \mid S \subseteq [\![\Phi'[\Gamma/X]]\!]e[Y \mapsto S] \,\} \\
&= \bigcup\{\, S \mid S \subseteq [\![\Phi']\!]e[Y \mapsto S][X \mapsto [\![\Gamma]\!]e[Y \mapsto S]] \,\} \\
&\qquad \text{by the induction hypothesis} \\
&= \bigcup\{\, S \mid S \subseteq [\![\Phi']\!]e[Y \mapsto S][X \mapsto [\![\Gamma]\!]e] \,\} \\
&\qquad \text{since } Y \text{ is not free in } \Gamma \\
&= [\![\Phi]\!]e[X \mapsto [\![\Gamma]\!]e].
\end{aligned}$$

□

Corollary 2.5 $[\![\nu X.\Phi]\!]e = [\![\Phi[\nu X.\Phi/X]]\!]e$.

The proposition $\nu X.\Phi$ may also be interpreted as an infinite conjunction when $\mathcal{S}$ is finite. In this case the complete lattice $\langle 2^{\mathcal{S}}, \subseteq, \cup, \cap \rangle$ is finite, and every monotonic function over this lattice is therefore continuous. The greatest and least fixed points of a continuous function ϕ over a complete lattice may be characterized as

$$\begin{aligned}
\nu\phi &= \bigcap_{i=0}^{\infty} \phi_i \\
\mu\phi &= \bigcup_{i=0}^{\infty} \hat{\phi}_i
\end{aligned}$$

where

$$\begin{aligned}
\phi_0 &= \mathcal{S} \\
\phi_{i+1} &= \phi(\phi_i) \\
\hat{\phi}_0 &= \emptyset \\
\hat{\phi}_{i+1} &= \phi(\hat{\phi}_i).
\end{aligned}$$

Now let *True* represent the proposition $\nu X.X$; clearly, $[\![\mathit{True}]\!]e = \mathcal{S}$ for any environment e. Using lemma 2.4 the proposition $\nu X.\Phi$ can be shown to be semantically equivalent to $\bigwedge_{i=0}^{\infty} \Phi_i$, where $\Phi_0 = \mathit{True}$ and $\Phi_{i+1} = \Phi[\Phi_i/X]$, while $\mu X.\Phi$ is equivalent to $\bigvee_{i=0}^{\infty} \hat{\Phi}_i$, where $\hat{\Phi}_0 = \neg\mathit{True}$ and $\hat{\Phi}_{i+1} = \Phi[\hat{\Phi}_i/X]$.

The expressiveness of this logic has been thoroughly analyzed, and interested readers are referred to [10, 11, 13, 18]. Examples of temporal logic operators expressed in the mu-calculus include the following (where $\mathit{Act} = \{a\}$).

$$\begin{aligned}
\mathit{Always}\ \Phi &= \nu X.(\Phi \wedge [a]X) \\
\mathit{Eventually}\ \Phi &= \mu X.(\Phi \vee (\langle a \rangle \mathit{True} \wedge [a]X))
\end{aligned}$$

R1 $$\boxed{\dfrac{H \vdash s \in \neg\neg\Phi}{H \vdash s \in \Phi}}$$

R2 $$\boxed{\dfrac{H \vdash s \in \Phi_1 \vee \Phi_2}{H \vdash s \in \Phi_1}}$$

R3 $$\boxed{\dfrac{H \vdash s \in \Phi_1 \vee \Phi_2}{H \vdash s \in \Phi_2}}$$

R4 $$\boxed{\dfrac{H \vdash s \in \neg(\Phi_1 \vee \Phi_2)}{H \vdash s \in \neg\Phi_1, H \vdash s \in \neg\Phi_2}}$$

R5 $$\boxed{\dfrac{H \vdash s \in \langle a \rangle \Phi}{H \vdash s' \in \Phi} \ (s' \in \{\, s' \mid s \xrightarrow{a} s' \,\})}$$

R6 $$\boxed{\dfrac{H \vdash s \in \neg\langle a \rangle \Phi}{H \vdash s_1 \in \neg\Phi, H \vdash s_2 \in \neg\Phi, \ldots} \ (\{s_1, s_2, \ldots\} = \{\, s' \mid s \xrightarrow{a} s' \,\})}$$

R7 $$\boxed{\dfrac{H \vdash s \in \nu X.\Phi}{H' \cup \{s : \nu X.\Phi\} \vdash s \in \Phi[\nu X.\Phi / X]} \ (s : \nu X.\Phi \notin H)}$$

R8 $$\boxed{\dfrac{H \vdash s \in \neg\nu X.\Phi}{H' \cup \{s : \nu X.\Phi\} \vdash s \in \neg\Phi[\nu X.\Phi / X]} \ (s : \nu X.\Phi \notin H)}$$

where $H' = H - \{\, s' : \Gamma \mid \nu X.\Phi \prec \Gamma \,\}$.

Figure 3: Tableau rules for the propositional mu-calculus.

3 The Tableau-Based Proof System

We now describe a proof system for establishing when states in a model M satisfy formulas in the mu-calculus. The proof rules operate on *sequents* of the form $H \vdash_M s \in \Phi$, where M is the model, s is a state from M, Φ is a formula and H is a set of *hypotheses*, or *assumptions*, of the form $s' : \Gamma$, for s' a state and Γ a *closed recursive formula*. In the remainder of this paper sequents will be ranged over by $\sigma, \ldots$, and references to the model M will be omitted. The intended meaning of a sequent $H \vdash s \in \Phi$ is that under assumptions H, s satisfies Φ. This notion is made precise in section 4.

Figure 3 lists the proof rules that we consider. The proof system is *tableau-based*, meaning that proofs are conducted in a top-down fashion; accordingly, the proof rules are written with conclusions appearing above premisses, as opposed to the more traditional style. It should be noted that we have written the rules this way to emphasize the fact that the proof system may serve as the basis of a decision procedure for determining whether states have properties; in fact, it is a simple matter to generate a more traditional, Post-style axiomatization. We shall say more about this later. The rules are also distinguished by their treatment of negated formulas and recursive formulas. Rules R4, R6 and R8 stipulate that negations be "driven inside" formulas, while R1 allows double negations to be eliminated. R7 and R8 require that in order to prove establish that a state enjoys a (negated) recursive property, it is sufficient to establish that it enjoys the (negated) unrolling of the property, provided that assumptions involving formulas having the recursive formula as a subformula are removed, or *discharged*, from the hypothesis list. The intuitive reason why this is necessary is the following. In the sequent $H \vdash s \in \nu X.\Phi$, assumptions involving $\nu X.\Phi$ as a subformula should play no role in determining whether, under H, s has property $\nu X.\Phi$, since these assumptions involve formulas which are *not* contained in $\nu X.\Phi$. However, when $\nu X.\Phi$ is unwound to $\Phi[\nu X.\Phi/X]$, some of these assumptions may involve subformulas of $\Phi[\nu X.\Phi/X]$ and may therefore improperly play a role in the proof of $H' \cup \{s : \nu X.\Phi\} \vdash s \in \Phi[\nu X.\Phi/X]$, and hence of $H \vdash s \in \nu X.\Phi$. To prevent this anomaly, then, such assumptions must be removed as $\nu X.\Phi$ is unwound.

Following [25], we shall say that a *tableau* for σ is a maximal proof tree having σ as its root and constructed using R1-R8. If σ' is a sequent resulting from the application of a rule to σ, then we say that σ' is a *child* of σ in the tableau, and that σ is the *parent* of σ'. A sequent in a tableau is a *leaf* if it has no children. The *height* of a tableau is defined as the length of the longest sequence $\langle \sigma_0, \sigma_1, \ldots \rangle$, where σ_i is the parent of σ_{i+1}.

A leaf $H \vdash s \in \Phi$ in a tableau is *successful* exactly when it meets one of the following conditions.

1. $\Phi \in \mathcal{A}$ and $s \in V(\Phi)$.

2. Φ is $\neg A$ for some $A \in \mathcal{A}$ and $s \notin V(A)$.

3. Φ is $\neg\langle a \rangle \Phi'$ for some a and Φ'.

4. Φ is $\nu X.\Phi'$ for some X and Φ'.

Notice that $H \vdash s \in \neg\langle a \rangle \Phi$ is a leaf only when s has no a-derivatives, while $H \vdash s \in \nu X.\Phi$ is a leaf only when $s : \nu X.\Phi \in H$. A *tableau* is successful exactly when all its leaves are successful; the intention is that a sequent σ has a proof if it has a successful tableau.

A more traditional proof system may be obtained as follows. Let the axioms be successful leaves and the rules of inference be the inverted forms of R1-R8. Then a successful (finite) tableau for σ in the tableau system corresponds to a proof of σ in this system.

It is also possible to develop inference rules based on R1-R8 for the derived operators presented in the previous section. Figure 4 contains a sample of such rules. They may easily be seen to follow from rules R1-R8; DR3 follows from R8, provided the assumption $s : \neg\mu X.\Phi$ in DR3 is translated as $s : \nu X.\neg\Phi[\neg X/X]$. It is also possible to derive rules for other constructs defined in the mu-calculus, such as the temporal operators *Always* and *Eventually*.

There are also the following derived success criteria for sequents involving *True* and $[a]$.

1. Sequents of the form $H \vdash s \in \mathit{True}$ are successful.

2. Leaves of the form $H \vdash s \in [a]\Phi$ are successful.

These follow from the fact that any sequent $H \vdash s \in \mathit{True}$ must have a successful tableau and from the definition of $[a]$.

In the remainder of this section we present an example of a tableau generated using rules R1-R8 and DR1-DR3. The property being tested asserts that, for transition systems $\langle \mathcal{S}, \mathit{Act}, \rightarrow \rangle$ with $\mathit{Act} = \{a, b\}$, and such that no state is *terminated* (i.e. has no transitions), it is always the case that an a action is infinitely often possible. The tableau is contained in

$$\text{DR1}\quad \boxed{\dfrac{H \vdash s \in \Phi_1 \wedge \Phi_2}{H \vdash s \in \Phi_1, H \vdash s \in \Phi_2}}$$

$$\text{DR2}\quad \boxed{\dfrac{H \vdash s \in [a]\Phi}{H \vdash s_1 \in \Phi, H \vdash s_2 \in \Phi, \ldots}\ (\{s_1, s_2, \ldots\} = \{\, s' \mid s \xrightarrow{a} s' \,\})}$$

$$\text{DR3}\quad \boxed{\dfrac{H \vdash s \in \mu X.\Phi}{H' \cup \{s : \neg\mu X.\Phi\} \vdash s \in \Phi[\mu X.\Phi/X]}\ (s : \neg\mu X.\Phi \notin H)}$$

$$H' \quad = \quad H - \{\, s' : \Gamma \mid \mu X.\Phi \prec \Gamma \,\}$$

Figure 4: Derived rules.

figure 5; it is successful, since each leaf is successful. An important thing to notice is that, if it were not for the discharging of assumptions involving $\neg B$ every time a new assumption for A is introduced, no successful tableau for for $\emptyset \vdash s \in A$ would exist, and the proof system would be unsound.

4 Soundness and Completeness

This section establishes the soundness and completeness for finite-state models of the proof system presented in the previous section. We first semantically characterize sequents by *relativizing* the semantics of propositions to hypothesis sets; we do so by defining a new semantic function, $[\![\Phi]\!]^H e$, for formulas Φ, environments e and hypothesis sets H. It will turn out that if H is empty, then $[\![\Phi]\!]^H e = [\![\Phi]\!] e$. We shall then prove theorems that establish the following, for closed Φ. (In the remainder of the paper, the parameter e to the semantic function will occasionally be omitted when we refer to the semantics of closed Φ, since in this case $[\![\Phi]\!] e = [\![\Phi]\!] e'$ for any environments e and e'.)

$$H \vdash s \in \Phi \textit{ has a successful tableau if and only if } s \in [\![\Phi]\!]^H.$$

We start by defining some notation and stating a simple lemma about monotonic functions.

Definition 4.1 *Let S' and $\mathcal{X}$ be sets, with $S' \subseteq \mathcal{X}$, and let ϕ be a monotonic function over the complete lattice $\langle 2^{\mathcal{X}}, \subseteq, \cup, \cap \rangle$. Then $\phi_{S'}$ is the function defined by*

$$\phi_{S'}(S) = \phi(S' \cup S).$$

Lemma 4.2 *Let $\mathcal{X}$ be a set, with $x \in \mathcal{X}$ and $S \subseteq \mathcal{X}$, and let ϕ be a monotonic function over the complete lattice $\langle 2^{\mathcal{X}}, \subseteq, \cup, \cap \rangle$. Then the following hold.*

1. *ϕ_S is monotonic.*

2. *$x \in \nu\phi$ if and only if $x \in \nu\phi_{\{x\}}$.*

Transition system:
$$\begin{tikzcd} s_1 \arrow[r, "a"] & s_2 \arrow[l, bend right, "b"'] \end{tikzcd}$$

For syntactic simplicity, the following abbreviations will be used. Note that B is the unrolling of A.

$$\begin{array}{rcl} A & \equiv & \nu X.(\mu Y.([a]X \wedge [b]Y)) \\ B & \equiv & \mu Y.([a]A \wedge [b]Y) \end{array}$$

The tableau below establishes that under no assumptions, s_1 has property A. In order to fit the tableau on the page, it is broken into two pieces.

$$\begin{array}{c} \emptyset \vdash s_1 \in A \\ \hline s_1 : A \vdash s_1 \in B \\ \hline s_1 : A, s_1 : \neg B \vdash s_1 \in [a]A \wedge [b]B \\ \hline \begin{array}{c|c} \begin{array}{c} s_1 : A, s_1 : \neg B \vdash s_1 \in [a]A \\ \hline s_1 : A, s_1 : \neg B \vdash s_2 \in A \\ \hline s_1 : A, s_2 : A \vdash s_2 \in B \\ \hline s_1 : A, s_2 : A, s_2 : \neg B \vdash s_2 \in [a]A \wedge [b]B \\ \hline \begin{array}{c|c} s_1 : A, s_2 : A, s_2 : \neg B \vdash s_2 \in [a]A & \text{See subtableau.} \end{array} \end{array} & s_1 : A, s_1 : \neg B \vdash s_1 \in [b]B \end{array} \end{array}$$

Subtableau:

$$\begin{array}{c} s_1 : A, s_2 : A, s_2 : \neg B \vdash s_2 \in [b]B \\ \hline s_1 : A, s_2 : A, s_2 : \neg B \vdash s_1 \in B \\ \hline s_1 : A, s_2 : A, s_1 : \neg B, s_2 : \neg B \vdash s_1 \in [a]A \wedge [b]B \\ \hline \begin{array}{c|c} \begin{array}{c} s_1 : A, s_2 : A, s_1 : \neg B, s_2 : \neg B \vdash s_1 \in [a]A \\ \hline s_1 : A, s_2 : A, s_1 : \neg B, s_2 : \neg B \vdash s_2 \in A \end{array} & s_1 : A, s_2 : A, s_1 : \neg B, s_2 : \neg B \vdash s_1 \in [b]B \end{array} \end{array}$$

Figure 5: A sample tableau.

3. Suppose that $x \in \nu\phi$. Then $\nu\phi = \nu\phi_{\{x\}}$.

Proof. The first of these follows directly from the definitions of monotonicity and $\phi_{S'}$. The proof of (2) breaks into two pieces.
($\Rightarrow$) Suppose that $x \in \nu\phi$. From the definition of $\nu\phi$, we have the following.

$$\begin{array}{rcl} x \in \nu\phi & \iff & x \in \bigcup\{\, X \subseteq \mathcal{X} \mid X \subseteq \phi(X)\,\} \\ & \iff & \exists X' \subseteq \mathcal{X}.\, x \in X' \wedge X' \subseteq \phi(X') \end{array}$$

Since $x \in X'$ it follows that $X' = \{x\} \cup X'$, whence $\phi(X') = \phi_{\{x\}}(X')$ and $X' \subseteq \phi_{\{x\}}(X')$. By definition, then, $x \in \nu\phi_{\{x\}}(X')$.
($\Leftarrow$) Suppose $x \in \nu\phi_{\{x\}}(X')$. This means that there is an $X \subseteq \mathcal{X}$ with $x \in X$ and $X \subseteq \phi_{\{x\}}(X)$. Since $\phi_{\{x\}}(X) = \phi(X \cup \{x\})$ it follows that $\phi_{\{x\}}(X) = \phi(X)$, and hence $x \in \nu\phi$.

To prove (3), suppose that $x \in \nu\phi$. Since $\nu\phi = \phi(\nu\phi)$ and $\nu\phi = \nu\phi \cup \{x\}$, it follows that

$$\nu\phi = \phi(\nu\phi \cup \{x\}) = \phi_{\{x\}}(\nu\phi),$$

and as $\nu\phi_{\{x\}}$ is the greatest fixed point of $\phi_{\{x\}}$ it follows that $\nu\phi \subseteq \nu\phi_{\{x\}}$. To see that $\nu\phi_{\{x\}} \subseteq \nu\phi$, observe the following.

$$\begin{array}{rcl} \nu\phi_{\{x\}} & = & \phi_{\{x\}}(\nu\phi_{\{x\}}) \\ & = & \phi(\nu\phi_{\{x\}} \cup \{x\}) \\ & = & \phi(\nu\phi_{\{x\}}) \;\; \text{since } x \in \nu\phi \text{ implies } x \in \nu\phi_{\{x\}} \text{ by (2)} \end{array}$$

Since $\nu\phi$ is the greatest fixed point of ϕ, the result follows. $\square$

The next definition will be useful in the remainder of the paper.

Definition 4.3 *Let H be a hypothesis set and Φ a formula. Then*

$$H\lceil\Phi = \{\, s \mid s : \Phi \in H \,\}.$$

From the definition of hypothesis sets, it follows that $H\lceil\Phi$ is nonempty only if Φ is a closed formula of the form $\nu X.\Phi'$.

The relativized semantics are given in figure 6. The only essential difference between this account and the one in figure 2 involves the fixed point operator. Here hypotheses $s : \Phi$ are interpreted as assertions that s "satisfies" Φ; accordingly, the set of states for such a formula includes all states assumed to satisfy the formula. These assumptions are also used to alter the function whose maximum fixed point forms the other component in the meaning of the formula, reflecting the fact that assumptions not only affect the meaning of the formula in question but also the meaning of the unrolling of the formula.

4.1 The Finiteness of Tableaux

In the remainder of the paper we shall restrict our attention to finite models, i.e. models $\langle \mathcal{S}, Act, \rightarrow, V \rangle$ where $|\mathcal{S}| < \infty$. Our goal in this subsection is to establish that for such models, every sequent σ has a maximum height tableau. This result enables us to prove

$$
\begin{aligned}
[\![A]\!]^H e &= V(A)\\
[\![X]\!]^H e &= e(X)\\
[\![\neg\Phi]\!]^H e &= \mathcal{S} - [\![\Phi]\!]^H e\\
[\![\Phi_1 \vee \Phi_2]\!]^H e &= [\![\Phi_1]\!]^H e \cup [\![\Phi_2]\!]^H e\\
[\![\langle a\rangle\Phi]\!]^H e &= \pi_a([\![\Phi]\!]^H e)\\
[\![\nu X.\Phi]\!]^H e &= (\nu\phi_{S'}) \cup S'
\end{aligned}
$$

$$
\begin{aligned}
\text{where}\quad \pi_a(S) &= \{\, s' \mid \exists s \in S.\ s' \xrightarrow{a} s \,\}\\
\phi(S) &= [\![\Phi]\!]^H e[X \mapsto S]\\
S' &= H\lceil \nu X.\Phi
\end{aligned}
$$

Figure 6: The relativized semantics of propositions.

results about tableaux for sequent σ by induction on the maximum height such tableaux may have; this proof technique is used in the proofs of soundness and completeness.

We start by defining an ordering $<$ on sequents that has no infinite ascending chains; the proof of the main result then proceeds by well-founded induction on the inverse of this relation. Intuitively, this relation is to hold between σ_1 and σ_2 if it is possible that σ_1 is an ancestor of σ_2 in some tableau; accordingly, it should be the case that if σ_1 is the parent of σ_2 in a tableau, then $\sigma_1 < \sigma_2$. This implies that the ordering should consist of two parts–one reflecting the fact that the application of certain rules (R1-R6) results in "shorter formulas", and one reflecting the fact that applications of other rules (R7 and R8) result in "larger" hypothesis sets.

To make all this precise, we first need to define an appropriate ordering on hypothesis sets. This requires several auxiliary notions, the first of which is the *closure* of a formula defined below.

Definition 4.4 *Let Φ be a formula. Then $CL(\Phi)$ is the smallest set satisfying the following.*

1. $\Phi \in CL(\Phi)$.
2. $\neg\neg\Phi' \in CL(\Phi) \Rightarrow \Phi' \in CL(\Phi)$.
3. $\Phi_1 \vee \Phi_2 \in CL(\Phi) \Rightarrow \Phi_1 \in CL(\Phi), \Phi_2 \in CL(\Phi)$.
4. $\neg(\Phi_1 \vee \Phi_2) \in CL(\Phi) \Rightarrow \neg\Phi_1 \in CL(\Phi), \neg\Phi_2 \in CL(\Phi)$.
5. $\langle a\rangle\Phi' \in CL(\Phi) \Rightarrow \Phi' \in CL(\Phi)$.
6. $\neg\langle a\rangle\Phi' \in CL(\Phi) \Rightarrow \neg\Phi' \in CL(\Phi)$.
7. $\nu X.\Phi' \in CL(\Phi) \Rightarrow \Phi'[\nu X.\Phi'/X] \in CL(\Phi)$.
8. $\neg\nu X.\Phi' \in CL(\Phi) \Rightarrow \neg\Phi'[\nu X.\Phi'/X] \in CL(\Phi)$.

The significance of $CL(\Phi)$ is that for any sequent $H' \vdash s' \in \Phi'$ appearing in a tableau constructed for $H \vdash s \in \Phi$, $\Phi' \in CL(\Phi)$. A similar notion of closure appears in [16], and as is pointed out there, $CL(\Phi)$ is finite, and indeed bounded in size by $|\Phi|$. Moreover, the following holds; we state it without proof.

Lemma 4.5 *Let Φ be a formula, and suppose that $\Phi' \in CL(\Phi)$. Then $CL(\Phi') \subseteq CL(\Phi)$.*

We now define a formula-indexed family of relations on hypothesis sets that will serve as the basis of the final relation on hypothesis sets. For formula Φ, $\sqsubseteq_\Phi$ is designed to relate hypothesis sets on the basis of the assumptions made about the closed recursive subformulas of Φ. Let $\mathcal{H}$ represent the set of all hypothesis sets.

Definition 4.6 *Let Φ be a formula. Then $\sqsubseteq_\Phi$ is defined by induction on the structure of Φ as follows.*

1. $\Phi \in \mathcal{A} \Rightarrow\ \sqsubseteq_\Phi = \mathcal{H} \times \mathcal{H}$.
2. $\Phi \in \mathcal{V} \Rightarrow\ \sqsubseteq_\Phi = \mathcal{H} \times \mathcal{H}$.
3. $\Phi = \neg\Phi' \Rightarrow\ \sqsubseteq_\Phi = \sqsubseteq_{\Phi'}$.
4. $\Phi = \Phi_1 \vee \Phi_2 \Rightarrow\ \sqsubseteq_\Phi = \sqsubseteq_{\Phi_1} \cap \sqsubseteq_{\Phi_2}$.
5. $\Phi = \langle a \rangle \Phi' \Rightarrow\ \sqsubseteq_\Phi = \sqsubseteq_{\Phi'}$.
6. $\Phi = \nu X.\Phi' \Rightarrow\ \sqsubseteq_\Phi = \{ \langle H_1, H_2 \rangle \in \sqsubseteq_{\Phi'} \mid \langle H_2, H_1 \rangle \in \sqsubseteq_{\Phi'}\ \Rightarrow H_1 \lceil \Phi \subseteq H_2 \lceil \Phi \}$.

We shall write $H_1 \sqsubseteq_\Phi H_2$ in lieu of $\langle H_1, H_2 \rangle \in \sqsubseteq_\Phi$. These relations are best understood via an example. Let $\mathcal{S} = \{s_1, s_2\}$, and consider

$$\Phi = \nu X.(\nu Y.((\nu Z.Z) \vee Y) \vee X).$$

Letting the variables X, Y and Z stand for their associated fixed point formulas, we have the following.

$$\begin{array}{rcl} \{s_1 : Y, s_1 : X\} & \sqsubseteq_\Phi & \{s_1 : Z\} \\ \{s_1 : Z, s_1 : Y, s_1 : X\} & \sqsubseteq_\Phi & \{s_1 : Z, s_1 : Y, s_2 : Y\} \end{array}$$

The next lemmas characterize some properties of $\sqsubseteq_\Phi$ that will be used in what follows.

Lemma 4.7 *Let Φ be a formula. Then $\sqsubseteq_\Phi$ is a preorder.*

Proof. We must show that $\sqsubseteq_\Phi$ is reflexive and transitive. Reflexivity is straightforward, and the proof is omitted. We establish the transitivity of $\sqsubseteq_\Phi$ by induction on the structure of Φ. The induction hypothesis states that for all $\Phi' \prec \Phi$, $\sqsubseteq_{\Phi'}$ is transitive. Most cases are routine; we consider here the case when Φ is $\nu X.\Phi'$. Suppose, then, that $H_1 \sqsubseteq_\Phi H_2 \sqsubseteq_\Phi H_3$; we must show that $H_1 \sqsubseteq_\Phi H_3$, i.e. that $H_1 \sqsubseteq_{\Phi'} H_3$ and that if $H_3 \sqsubseteq_{\Phi'} H_1$ then $H_1 \lceil \Phi \subseteq H_3 \lceil \Phi$. The former follows from the induction hypothesis and the fact that $\sqsubseteq_\Phi \subseteq \sqsubseteq_{\Phi'}$. To prove the latter, assume that $H_3 \sqsubseteq_{\Phi'} H_1$; we must show that $H_1 \lceil \Phi \subseteq H_3 \lceil \Phi$. It is sufficient to show that $H_3 \sqsubseteq_{\Phi'} H_2$ and $H_2 \sqsubseteq_{\Phi'} H_1$, since these facts, coupled with the facts that $H_2 \sqsubseteq_\Phi H_3$ and $H_1 \sqsubseteq_\Phi H_2$, allow us to conclude that $H_1\lceil\Phi \subseteq H_2\lceil\Phi \subseteq H_3\lceil\Phi$, and hence that $H_1\lceil\Phi \subseteq H_3\lceil\Phi$. Since $H_1 \sqsubseteq_\Phi H_2 \sqsubseteq_\Phi H_3$, the definition of $\sqsubseteq_\Phi$ ensures that $H_1 \sqsubseteq_{\Phi'} H_2 \sqsubseteq_{\Phi'} H_3$, and as $H_3 \sqsubseteq_{\Phi'} H_1$ by assumption, we have that $H_1 \sqsubseteq_{\Phi'} H_2 \sqsubseteq_{\Phi'} H_3 \sqsubseteq_{\Phi'} H_1$. But the induction hypothesis guarantees that $\sqsubseteq_{\Phi'}$ is transitive, and therefore we have that $H_3 \sqsubseteq_{\Phi'} H_2$ and $H_2 \sqsubseteq_{\Phi'} H_1$. □

On the basis of this lemma, we can define the equivalence relation, $=_\Phi$, and the strict ordering relation, $\sqsubset_\Phi$, associated with Φ.

Definition 4.8 *Let H_1 and H_2 be hypothesis sets.*

1. *$H_1 =_\Phi H_2$ exactly when $H_1 \sqsubseteq_\Phi H_2$ and $H_2 \sqsubseteq_\Phi H_1$.*
2. *$H_1 \sqsubset_\Phi H_2$ exactly when $H_1 \sqsubseteq_\Phi H_2$ and $H_2 \not\sqsubseteq_\Phi H_1$.*

It also turns out that the hypothesis sets generated by R7 (and R8) are "larger" (according to the recursive formula in the sequent the rule is applied to) than the hypothesis set in the sequent the rule is applied to.

Lemma 4.9 *Let $\nu X.\Phi$ be a formula and H and H' hypothesis sets with*

$$H' = H - \{\, s' : \Gamma \mid \nu X.\Phi \prec \Gamma \,\}.$$

Then $H \sqsubset_{\nu X.\Phi} H' \cup \{s : \nu X.\Phi\}$ if $s : \nu X.\Phi \notin H$.

Proof. Since for any $\Gamma \prec \nu X.\Phi$, $H\lceil\Gamma = H'\lceil\Gamma$, it follows that $H' =_\Phi H$, and as $H\lceil\nu X.\Phi \subset (H'\lceil\nu X.\Phi) \cup \{s\}$, it is the case that $H \sqsubset_{\nu X.\Phi} H'$. □

Moreover, in a certain sense substitution "preserves" $\sqsubset_\Phi$.

Lemma 4.10 *Suppose that $H_1 \sqsubset_\Phi H_2$, $H_1 \sqsubseteq_\Gamma H_2$, and Y is free in Γ. Then $H_1 \sqsubset_{\Gamma[\Phi/Y]} H_2$.*

Proof. The proof proceeds by induction on Γ. Most cases are straightforward; we consider here the case when $\Gamma \equiv \nu X.\Gamma'$. The induction hypothesis states that if $H_1 \sqsubseteq_{\Gamma'} H_2$ then $H_1 \sqsubset_{\Gamma'[\Phi/Y]} H_2$. From the definition of $\sqsubseteq_\Gamma$ it follows that $H_1 \sqsubseteq_{\Gamma'} H_2$, and the induction hypothesis guarantees that $H_1 \sqsubset_{\Gamma'[\Phi/Y]} H_2$. Since it cannot be the case, then, that $H_1 =_{\Gamma'[\Phi/Y]} H_2$, it is by definition the case that $H_1 \sqsubset_{\Gamma[\Phi/Y]} H_2$. □

We now prove that $\sqsubset_\Phi$ has no infinite ascending chains.

Lemma 4.11 *Let Φ be a formula. Then $\sqsubset_\Phi$ has no infinite ascending chains.*

Proof. By induction on the structure of Φ. Most cases are routine; we consider here the case when Φ is $\nu X.\Phi'$. The induction hypothesis states that $\sqsubset_{\Phi'}$ has no ascending chains. From the definitions of $\sqsubseteq_\Phi$ and $\sqsubset_\Phi$ we can deduce the following.

$$\begin{aligned} \sqsubset_\Phi &= \sqsubset_{\Phi'} \cup R, \quad \text{where} \\ R &= \{\, \langle H_1, H_2\rangle \mid H_1 =_{\Phi'} H_2 \wedge H_1\lceil\Phi \subset H_2\lceil\Phi \,\} \end{aligned}$$

The relation R clearly has no infinite ascending chains, since $|\mathcal{S}| < \infty$ and $H\lceil\Phi \subseteq \mathcal{S}$ for all H. It is also transitive, since $=_{\Phi'}$ is an equivalence relation and $\subset$ is transitive. To establish that the $\sqsubset_\Phi$ has no infinite ascending chains, assume by way of contradiction that $C = \langle H_0, H_1, \ldots\rangle$ is an infinite ascending $\sqsubset_\Phi$-chain of hypothesis sets. Without loss of generality we may further assume that $H_0 \sqsubset_{\Phi'} H_1$. We can now form the following collection of contiguous segments of C,

$$\begin{aligned} C_0 &= \langle H_0, H_1, \ldots, H_{i_0}\rangle \\ C_1 &= \langle H_{i_0}, H_{i_0+1}, \ldots, H_{i_1}\rangle \\ &\vdots \\ C_j &= \langle H_{i_{j-1}}, H_{i_{j-1}+1}, \ldots, H_{i_j}\rangle \\ &\vdots \end{aligned}$$

where the C_j satisfy the following "maximality" conditions: each C_{2j} is a $\sqsubset_{\Phi'}$-chain, $H_{i_{2j}} \not\sqsubset_{\Phi'} H_{i_{2j}+1}$, each C_{2j+1} is an R-chain, and $H_{i_{2j+1}} \not R\, H_{i_{2j+1}+1}$. Notice that these imply that each C_j has at least two elements and that each C_j is finite. Now consider the sequence $H_0, H_{i_2}, \ldots$, comprising the first elements of each C_{2j}. This sequence is infinite; moreover, given the definition of R, the fact that $\sqsubseteq_{\Phi'}$ is transitive, and the fact that each C_i contains at least two elements, it follows that $H_0 \sqsubset_{\Phi'} H_{i_2} \sqsubset_{\Phi'} \ldots$, which contradicts the induction hypothesis, namely, that $\sqsubset_{\Phi'}$ contains no infinite ascending chains. Therefore, $\sqsubset_\Phi$ contains no infinite ascending chains. $\square$

Recall that our goal is to define an ordering relation on hypothesis sets such that the hypothesis sets generated by applying a rule to a sequent are "at least" as large as the hypothesis set in the sequent. We define such a relation on the basis of $\sqsubseteq_\Phi$ and $CL(\Phi)$.

Definition 4.12 *Let $H_1, H_2 \in \mathcal{H}$. Then $H_1 \trianglelefteq_\Phi H_2$ holds exactly when for every $\Phi' \in CL(\Phi)$, $H_1 \sqsubseteq_{\Phi'} H_2$.*

Thus, if $H_1 \trianglelefteq_\Phi H_2$ then $H_1 \sqsubseteq_{\Phi'} H_2$ for *any* Φ' that may appear in a tableau generated for Φ. It is clear that $\trianglelefteq_\Phi$ is a preorder; let $\equiv_\Phi$ and $\triangleleft_\Phi$ be the corresponding equivalence relation and strict ordering relation, respectively. Notice that lemma 4.11 and the fact that $CL(\Phi)$ is finite also guarantee that $\triangleleft_\Phi$ has no infinite ascending chains. We are now able to define the desired ordering on sequents and prove that it has no infinite ascending chains.

Definition 4.13 *Let σ_1 and σ_2 be sequents $H_1 \vdash s_1 \in \Phi_1$ and $H_2 \vdash s_2 \in \Phi_2$, respectively. Define $\sigma_1 < \sigma_2$ to hold when $\Phi_2 \in CL(\Phi_1)$ and one of the following is true.*

1. *$H_1 \triangleleft_{\Phi_2} H_2$, or*
2. *$H_1 \equiv_{\Phi_2} H_2$ and $|\Phi_1| > |\Phi_2|$.*

Lemma 4.14 *$<$ has no infinite ascending chains.*

Proof. Suppose there exists an infinite chain $\sigma_0 < \sigma_1 < \ldots$, where each σ_i has form $H_i \vdash s_i \in \Phi_i$. From the definition of $<$ it follows that $\Phi_{i+1} \in CL(\Phi_i)$, whence, by lemma 4.5, $CL(\Phi_{i+1}) \subseteq CL(\Phi_i) \subseteq CL(\Phi_0)$. Since $|CL(\Phi_0)| < \infty$, there must be a j with $CL(\Phi_k) =$

$CL(\Phi_j)$ for all $k \geq j$. This, however, leads to a contradiction, for in the infinite chain $\sigma_j < \sigma_{j+1} < \ldots$, $\sigma_k < \sigma_{k+1}$ exactly when $H_k \lhd_{\Phi_j} H_{k+1}$, or when $H_k \equiv_{\Phi_j} H_{k+1}$ and $|\Phi_1| > |\Phi_2|$, and these facts imply the existence of an infinite ascending $\lhd_{\Phi_j}$-chain. Therefore, $<$ has no infinite ascending chains. $\Box$

So the inverse of $<$ is well-founded, and we are now able to prove the next result by well-founded induction on this relation.

Theorem 4.15 *For any sequent $\sigma \equiv H \vdash s \in \Phi$, there is a maximum height tableau with root σ.*

Proof. Let σ be the sequent $H \vdash s \in \Phi$. The proof proceeds by well-founded induction on the inverse of $<$; the induction hypothesis states that for all σ' with $\sigma < \sigma'$, there is a maximum height tableau with σ' at its root. We now perform a case analysis on Φ; in each case we show that the result of applying any tableau rule results in a finite number of sequents σ' with $\sigma < \sigma'$ and then obtain the desired result on the basis of this fact and the induction hypothesis. Most cases are straightforward; we consider here the case when Φ is $\nu X.\Phi'$ and $s : \Phi \notin H$. The only applicable rule is R7, and the sequent resulting from the application of this rule is $\sigma' \equiv H' \cup \{s : \Phi\} \vdash s \in \Phi'[\Phi/X]$, where $H' = H - \{\, s : \Gamma \mid \Phi \prec \Gamma \,\}$. It remains for us to show that $\sigma < \sigma'$; then the induction hypothesis allows us to conclude the desired result. There are two cases to consider.

1. X is not free in Φ'. Then $\Phi'[\Phi/X]$ is Φ'; thus, if we prove that $H \equiv_{\Phi'} H' \cup \{s : \Phi\}$, then $\sigma < \sigma'$ follows from the fact that $|\Phi| > |\Phi'|$. Since $\Phi \not\prec \Phi'$, there cannot be a $\Gamma \in CL(\Phi')$ with $\Phi \prec \Gamma$; this implies that for all $\Gamma \in CL(\Phi')$, $H\lceil\Gamma = (H' \cup \{s : \Phi\})\lceil\Gamma$. From this fact, it follows that for any $\Gamma \in CL(\Phi')$, $H =_\Gamma (H' \cup \{s : \Phi\})$, and hence $H \equiv_{\Phi'} (H' \cup \{s : \Phi\})$.

2. X is free in Φ'. In this case it is easy to establish that $CL(\Phi) = CL(\Phi'[\Phi/X])$. It is therefore sufficient to show that $H \lhd_\Phi H'' = H' \cup \{s : \Phi\}$. By lemma 4.9 we have that $H \sqsubset_\Phi H''$; therefore, to establish that $H \lhd_\Phi H''$ we only need show that for all $\Gamma \in CL(\Phi)$, $H \sqsubseteq_\Gamma H''$. Let $\Gamma \in CL(\Phi)$. If $\Phi \not\preceq \Gamma$ then the result follows from the fact that for all $\Gamma' \prec \Gamma$, $H\lceil\Gamma' = H''\lceil\Gamma'$. If $\Phi \preceq \Gamma$, then let Γ' be the formula in $CL(\Phi')$ such that Γ is $\Gamma'[\Phi/X]$; this is guaranteed to exist from the definition of $CL(\Phi)$. It is straightforward to verify that $H \sqsubseteq_{\Gamma'} H''$, whence by lemma 4.10, $H \sqsubset_\Gamma H''$.

$\Box$

We can also prove the following, which implies that it is possible to enumerate all tableaux rooted by σ.

Theorem 4.16 *For any σ, there are a finite number of distinct tableaux with root σ.*

Proof. By induction on the height of the maximum height tableau for σ. $\Box$

4.2 Soundness and Completeness

In the remainder of the paper, we shall use $\mathcal{M}(\sigma)$ to refer to the maximum height of tableaux for σ and $\mathcal{M}_s(\sigma)$ to represent the maximum height of successful tableaux for σ (provided a successful tableau exists). In this section, we prove the soundness and completeness of the tableau proof system. The next lemma establishes a link needed in the proof of soundness between substitution (on the syntactic side) and function application (on the semantic side).

Lemma 4.17 *Suppose Φ is a formula, X a variable, and Γ a closed formula. Let H be a hypothesis set containing no hypotheses involving Γ as a strict subformula. Then*

$$[\![\Phi[\Gamma/X]]\!]^H e = \phi([\![\Gamma]\!]^H e),$$

where $\phi(S) = [\![\Phi]\!]^H e[X \mapsto S]$.

Proof. If X is not free in Φ then the result follows trivially. So assume that X is free in Φ; the proof proceeds by structural induction on Φ. Most cases are routine; we consider here the case when Φ is $\nu Y.\Phi'$. Given the definition of substitution we may assume that X and Y are distinct. Since Γ is closed we have that:

$$\begin{aligned} \Phi[\Gamma/X] &= \nu Y.(\Phi'[\Gamma/X]) \text{ whence} \\ [\![\Phi[\Gamma/X]]\!]^H e &= [\![\nu Y.(\Phi'[\Gamma/X])]\!]^H e \\ &= \nu\phi^{\Gamma}_{S'} \cup S' \end{aligned}$$

where

$$\begin{aligned} \phi^{\Gamma}(S) &= [\![\Phi'[\Gamma/X]]\!]^H e[Y \mapsto S] \text{ and} \\ S' &= H\lceil(\Phi[\Gamma/X]). \end{aligned}$$

Since X is free in Φ, $\Gamma \prec \Phi[\Gamma/X]$, so by assumption this implies that $S' = \emptyset$. Therefore,

$$[\![\Phi[\Gamma/X]]\!]^H e = \nu\phi^{\Gamma}.$$

Using the induction hypothesis and the fact that Γ is closed and X and Y are distinct, we have that for any S,

$$\begin{aligned} \phi^{\Gamma}(S) &= [\![\Phi'[\Gamma/X]]\!]^H e[Y \mapsto S] \\ &= [\![\Phi']\!]^H e[Y \mapsto S][X \mapsto [\![\Gamma]\!]^H e[Y \mapsto S]] \\ &= [\![\Phi']\!]^H e[X \mapsto [\![\Gamma]\!]^H e][Y \mapsto S]. \end{aligned}$$

Now consider the definition of $[\![\Phi]\!]^H e[X \mapsto [\![\Gamma]\!]^H e]$.

$$\begin{aligned} [\![\Phi]\!]^H e[X \mapsto [\![\Gamma]\!]^H e] &= \nu\phi_{S''} \cup S'', \text{ where} \\ \phi(S) &= [\![\Phi']\!]^H e[X \mapsto [\![\Gamma]\!]^H e][Y \mapsto S] \\ S'' &= H\lceil\Phi \end{aligned}$$

Since Φ has X free and H consists of assumptions involving only closed formulas, $S'' = \emptyset$; therefore, since $\phi(S) = \phi^{\Gamma}(S)$ for all S, it follows that $\nu\phi = \nu\phi^{\Gamma}$, and we are done. $\square$

We now have the necessary machinery to prove the soundness theorem for finite models.

Theorem 4.18 *If $H \vdash s \in \Phi$ has a successful tableau then $s \in [\![\Phi]\!]^H$.*

Proof. In light of the fact that every sequent has a maximum height tableau, it suffices to establish that every successful leaf is semantically valid and that each inference rule preserves soundness. The interesting cases involve the inference rules for fixed point formulas. We consider the proof of the validity of R7 in detail; the proof for R8 follows much the same lines.

Assume that H and H' are such that

$$H' = H - \{ s' : \Gamma \mid \nu X.\Phi \prec \Gamma \}$$

and $s \in [\![\Phi[\nu X.\Phi/X]]\!]^{(H' \cup \{s:\nu X.\Phi\})}$; we show that this implies that $s \in \nu X.\Phi$. By lemma 4.17, it follows that

$$[\![\Phi[\nu X.\Phi/X]]\!]^{(H' \cup \{s:\nu X.\Phi\})} e = \phi([\![\nu X.\Phi]\!]^{(H' \cup \{s:\nu X.\Phi\})}),$$

where $\phi(S) = [\![\Phi]\!]^{(H' \cup \{s:\nu X.\Phi\})} e[X \mapsto S]$ for some environment e. Now let $S' = H' \lceil \nu X.\Phi$. From the definition of the relativized semantics, it follows that

$$[\![\nu X.\Phi]\!]^{(H' \cup \{s:\nu X.\Phi\})} = \nu\phi_{S' \cup \{s\}} \cup S' \cup \{s\},$$

and therefore we have that

$$[\![\Phi[\nu X.\Phi/X]]\!]^{(H' \cup \{s:\nu X.\Phi\})} e = \phi(\nu\phi_{S' \cup \{s\}} \cup S' \cup \{s\}) = \phi_{S' \cup \{s\}}(\nu\phi_{S' \cup \{s\}}) = \nu\phi_{S' \cup \{s\}}$$

By lemma 4.2(2), since $s \in \nu\phi_{S' \cup \{s\}}$ we have that $s \in \nu\phi_{S'}$, and the desired result follows from the fact that $[\![\nu X.\Phi]\!]^H$ can be shown to be equal to $\nu\phi_{S'} \cup S'$. $\Box$

Completeness turns out to be a corollary of the soundness theorem and the next theorem.

Theorem 4.19 *$H \vdash s \in \Phi$ has a successful tableau if and only if $H \vdash s \in \neg\Phi$ has no successful tableau.*

Proof. ($\Rightarrow$) Suppose that $H \vdash s \in \Phi$ has a successful tableau. We prove that $H \vdash s \in \neg\Phi$ has no successful tableau by induction on $m = \mathcal{M}_s(H \vdash s \in \Phi)$.
BASE. In this case $m = 1$, and $H \vdash s \in \Phi$ is a successful leaf. The proof now proceeds by case analysis on the definition of a successful leaf. Most of the cases are straightforward; we show here the case when Φ is $\neg\langle a \rangle \Phi'$. For the sequent to be a leaf, it must be the case that $\{ s' \mid s \xrightarrow{a} s' \} = \emptyset$, and this implies that $H \vdash s \in \langle a \rangle \Phi'$ is an unsuccessful leaf. Therefore, $H \vdash s \in \neg\neg\langle a \rangle \Phi'$ can have no successful tableaux.
INDUCTION. Assume that $m > 1$. The induction hypothesis states that for all sequents $\sigma' \equiv H' \vdash s' \in \Phi'$ having a successful tableau with $\mathcal{M}_s(\sigma') < m$, $H' \vdash s' \in \neg\Phi'$ has no successful tableau. We proceed by a case analysis on Φ. Most cases are routine; we consider here two of them.

- Φ is $\nu X.\Phi'$. $H \vdash s \in \Phi$ has a successful tableau exactly when

  $$\sigma' \equiv H' \cup \{s : \Phi\} \vdash s \in \Phi'[\Phi/X]$$

  does, where $H' = H - \{ s' : \Gamma \mid \Phi \prec \Gamma \}$; moreover, it must be the case that $\mathcal{M}_s(\sigma') = m - 1$, whence by the induction hypothesis, $H' \cup \{s : \Phi\} \vdash s \in \neg\Phi'[\Phi/X]$ has no successful tableau. It therefore follows that $H \vdash s \in \neg\Phi$ has no successful tableau.

- Φ is $\neg\nu X.\Phi'$. In this case $H \vdash s \in \Phi$ has a successful tableau exactly when

  $$\sigma' \equiv H' \cup \{s : \Phi\} \vdash s \in \neg\Phi'[\Phi/X]$$

  does, where H' is as defined above. Moreover, since it must be the case that $\mathcal{M}_s(\sigma') = m - 1$, it follows from the induction hypothesis that $H' \cup \{s : \Phi\} \vdash s \in \neg\neg\Phi'[\Phi/X]$ has no successful tableau, and therefore $H' \cup \{s : \Phi\} \vdash s \in \Phi'[\Phi/X]$, $H \vdash s \in \nu X.\Phi'$, and $H \vdash s \in \neg\Phi$ do not, either.

($\Leftarrow$) We shall actually prove that if $\sigma \equiv H \vdash s \in \Phi$ has no successful tableau then $H \vdash s \in \neg\Phi$ must. We do so by induction on $m = \mathcal{M}(\sigma)$.
BASE. $m = 1$, meaning that σ is an unsuccessful leaf. The proof proceeds by a case analysis. The cases are routine; here we consider the case when Φ is $\langle a\rangle\Phi'$. For this to be a leaf (and therefore unsuccessful), it must be that $\{\, s' \mid s \xrightarrow{a} s' \,\} = \emptyset$, and therefore $H \vdash s \in \neg\Phi$ is by definition a successful leaf.
INDUCTION. Assume that $m > 1$. The induction hypothesis states that for all sequents $\sigma' \equiv H' \vdash s' \in \Phi'$ such that $\mathcal{M}(\sigma') < m$, if σ' has no successful tableau then $H' \vdash s' \in \neg\Phi'$ must. The proof proceeds by a case analysis on Φ. Most cases are routine; we show here two of them.

- Φ is $\nu X.\Phi'$. Since σ has no successful tableau, $H' \cup \{s : \Phi\} \vdash s \in \Phi'[\Phi/X]$ must have no successful tableau, and we can apply the induction hypothesis to conclude that $H' \cup \{s : \Phi\} \vdash s \in \neg\Phi'[\Phi/X]$ has a successful tableau. But this implies that $H \vdash s \in \neg\Phi$ has a successful tableau as well.

- Φ is $\neg\nu X.\Phi'$. Since σ has no successful tableau, $H' \cup \{s : \Phi\} \vdash s \in \neg\Phi'[\Phi/X]$ must have no successful tableau, and we can apply the induction hypothesis to conclude that $H' \cup \{s : \Phi\} \vdash s \in \neg\neg\Phi'[\Phi/X]$ must have a successful tableau. But this implies that $H' \cup \{s : \Phi\} \vdash s \in \Phi'[\Phi/X]$, and hence $H \vdash s \in \nu X.\Phi'$ and $H \vdash s \in \neg\Phi$, must also have successful tableaux.

□

Corollary 4.20 *If $s \in [\![\Phi]\!]^H$ then $H \vdash s \in \Phi$ has a successful tableau.*

Proof. Suppose to the contrary that $s \in [\![\Phi]\!]^H$ but that $H \vdash s \in \Phi$ has no successful tableau. By the previous theorem, it follows that $H \vdash s \in \neg\Phi$ has a successful tableau, and by the soundness theorem it follows that $s \in [\![\neg\Phi]\!]^H$, meaning that $s \notin [\![\Phi]\!]^H$, which is a contradiction. □

5 Towards an Efficient Algorithm

It is beyond the scope of this paper to pursue fully the question of an efficient tableau-based model checking algorithm for the mu-calculus. In this section, however, we present a straightforward procedure based on the results of the preceding sections and prove lemmas that enable information computed in one phase of the tableau construction process to be

```
fun check1' (H ⊢ s ∈ Φ) =
    case Φ is
    A ∈ 𝒜   ⟶ return(s ∈ V(A))
    X ∈ 𝒱   ⟶ error
    ¬Φ'     ⟶ return not (check1'(H ⊢ s ∈ Φ'))
    Φ1 ∨ Φ2 ⟶ return (check1'(H ⊢ s ∈ Φ1) or check1'(H ⊢ s ∈ Φ2))
    ⟨a⟩Φ'   ⟶ for each s' ∈ { s' | s -a-> s' } do
                    if check1'(H ⊢ s' ∈ Φ) then return true;
                return false
    νX.Φ'   ⟶ let H' = { s' : Γ | Φ ⊀ Γ } in
                    return (check1'(H' ∪ {s : Φ} ⊢ s ∈ Φ'[Φ/X]))
end

fun check1 (s ∈ Φ) = check1'(∅ ⊢ s ∈ Φ)
```

Figure 7: A simple model checking algorithm.

reused in others. The results presented in this section underlie an ongoing implementation of a model checker in the Concurrency Workbench [8, 9].

Figure 7 contains a basic algorithm for determining whether or not a state (in a finite model) satisfies a mu-calculus proposition. It is based on tableau rules R2, R3, R5 and R7, together with theorem 4.19; termination and correctness follow from the the results established in section 4. However, this algorithm is very inefficient. Although we shall not formally analyze its complexity here, it is worth noting that it exhibits exponential behavior even for formulas having no recursive subformulas, owing to the possibility of nested modal operators. The algorithm in [13] is also exponential in the worst case; however, it is linear time for formulas having no fixed points, and it runs in polynomial time for formulas resulting from the encodings of a wide variety logics in the mu-calculus.

The inefficiency of *check1* stems partially from the fact that no provision is made for the storing of results of sequents whose truth has already been determined. Accordingly, its performance can be improved somewhat by saving the results of previous computations and performing a lookup before recursively examining a sequent. Additionally, there are results that can be proved that allow the truth of sequents to be deduced solely on the basis of the truth of other sequents, and we devote the rest of this section to presenting some of them. The next lemma establishes that if an assumption turns out to be "true" with respect to a given hypothesis list, then it may be removed as an assumption without affecting the truth or falsity of the sequent.

Lemma 5.1 *Suppose that $H \vdash s \in \nu X.\Phi$ has a successful tableau. Then $H \cup \{s : \nu X.\Phi\} \vdash s' \in \Gamma$ has a successful tableau if and only if $H \vdash s' \in \Gamma$ does.*

Proof. In light of the soundness and completeness of the tableau system, it suffices to show that

$$[\![\Gamma]\!]^{H} e = [\![\Gamma]\!]^{H \cup \{s:\nu X.\Phi\}} e$$

under the assumption that $s \in [\![\nu X.\Phi]\!]^H e$. From the definition of the relativized semantics we have that

$$[\![\nu X.\Phi]\!]^H e = \nu\phi_{S'} \cup S',$$

where $\phi(S) = [\![\Phi]\!]^H e[X \mapsto S]$ and $S' = H\lceil\nu X.\Phi$. If $s : \nu X.\Phi \in H$ then $H \cup \{s : \nu X.\Phi\} = H$ and the result follows. So assume that $s : \nu X.\Phi \notin H$. Then $s \in \nu\phi_{S'}$, and by lemma 4.2(3) it follows that

$$\nu\phi_{S'} = \nu\phi_{S' \cup \{s\}},$$

whence $[\![\nu X.\Phi]\!]^H e = [\![\nu X.\Phi]\!]^{H \cup \{s:\nu X.\Phi\}} e$. The result now follows by a straightforward induction on Γ. $\Box$

This result can be used to "speed up" model checking as follows. If $H \cup \{s : \nu X.\Phi\} \vdash s' \in \Gamma$ has been shown to be true (false), and $H \vdash s \in \nu X.\Phi$ is subsequently shown to be true, then it is possible to deduce that $H \vdash s' \in \Gamma$ is true (false) without having to construct a tableau for it.

The next result establishes certain "monotonicity conditions" on hypothesis sets. We need the following definitions.

Definition 5.2 *The relations $\prec_-$ and $\preceq_+$ are defined inductively as follows.*

1. $\Phi \preceq_+ \Phi$ *for any* Φ.
2. $\Phi \preceq_+ \Phi' \Rightarrow \Phi \prec_- \neg\Phi'$.
3. $\Phi \prec_- \Phi' \Rightarrow \Phi \preceq_+ \neg\Phi'$.
4. $\Phi \preceq_+ \Phi_1 \Rightarrow \Phi \preceq_+ \Phi_1 \vee \Phi_2,\ \Phi \preceq_+ \Phi_2 \vee \Phi_1,\ \Phi \preceq_+ \langle a\rangle\Phi_1,\ \Phi \preceq_+ \nu X.\Phi_1$.
5. $\Phi \prec_- \Phi_1 \Rightarrow \Phi \prec_- \Phi_1 \vee \Phi_2,\ \Phi \prec_- \Phi_2 \vee \Phi_1,\ \Phi \prec_- \langle a\rangle\Phi_1,\ \Phi \prec_- \nu X.\Phi_1$.

Intuitively, $\Phi \preceq_+ \Gamma$ holds if there is an instance of Φ in Γ that occurs inside an even number of negations, while $\Phi \prec_- \Gamma$ holds if there is an instance of Φ in Γ that occurs inside an odd number of negations. From the definition, it is apparent that $\preceq_+ \cup \prec_- = \preceq$; it is also the case that both $\Phi \preceq_+ \Gamma$ and $\Phi \prec_- \Gamma$ hold for certain Φ and Γ.

Definition 5.3 *Let Φ be a formula, and let H_1 and H_2 be hypothesis sets. Then define $H_1 \subseteq_\Phi H_2$ to hold exactly when the following are true.*

1. *For each $\Gamma \preceq_+ \Phi$, $H_1\lceil\Gamma \subseteq H_2\lceil\Gamma$.*
2. *For each $\Gamma \prec_- \Phi$, $H_2\lceil\Gamma \subseteq H_1\lceil\Gamma$.*

Notice that the definition implies that if $H_1 \subseteq_\Phi H_2$, $\Gamma \preceq_+ \Phi$ and $\Gamma \prec_- \Phi$, then $H_1\lceil\Gamma = H_2\lceil\Gamma$.

The relation $\subseteq_\Phi$ defines a sort of "syntactic monotonicity" condition for hypothesis sets relative to Φ. Therefore, it should be the case that $H_1 \subseteq_\Phi H_2$ implies that $[\![\Phi]\!]^{H_1} e \subseteq [\![\Phi]\!]^{H_2} e$. This is in fact the case.

Lemma 5.4 *Let Φ be a formula, e an environment and H_1 and H_2 hypothesis sets with $H_1 \subseteq_\Phi H_2$. Then $[\![\Phi]\!]^{H_1} e \subseteq [\![\Phi]\!]^{H_2} e$.*

Proof. The proof proceeds by induction on Φ. There are several cases to consider, most of which are routine. We consider two of them here.

- $\Phi \equiv \neg\Phi'$. The induction hypothesis states that for all e and H_1 and H_2 with $H_1 \subseteq_{\Phi'} H_2$, $[\![\Phi']\!]^{H_1}e \subseteq [\![\Phi']\!]^{H_2}e$. Now suppose that $H_1 \subseteq_{\Phi} H_2$. From the definitions of $\subseteq_{\Phi}$, $\preceq_+$ and $\prec_-$, it follows that $H_2 \subseteq_{\Phi'} H_1$, and from the induction hypothesis we may conclude that $[\![\Phi']\!]^{H_2}e \subseteq [\![\Phi']\!]^{H_1}e$. Accordingly, $[\![\Phi]\!]^{H_1}e \subseteq [\![\Phi]\!]^{H_2}e$.

- $\Phi \equiv \nu X.\Phi'$. The induction hypothesis states that for all e, H_1 and H_2 with $H_1 \subseteq_{\Phi'} H_2$, $[\![\Phi']\!]^{H_1}e \subseteq [\![\Phi']\!]^{H_2}e$. Now suppose that $H_1 \subseteq_{\Phi} H_2$. It clearly follows that $H_1 \subseteq_{\Phi'} H_2$, and thus for any e, $[\![\Phi']\!]^{H_1}e \subseteq [\![\Phi']\!]^{H_2}e$. By definition, we have the following:

$$\begin{aligned}
[\![\Phi]\!]^{H_1} &= \nu\phi^1_{S_1} \cup S_1 \\
[\![\Phi]\!]^{H_2} &= \nu\phi^2_{S_2} \cup S_2 \;\text{ where} \\
\phi^1(S) &= [\![\Phi']\!]^{H_1}e[X \mapsto S] \\
\phi^2(S) &= [\![\Phi']\!]^{H_2}e[X \mapsto S] \\
S_1 &= H_1\lceil\Phi \\
S_2 &= H_2\lceil\Phi.
\end{aligned}$$

  Since $\Phi \preceq_+ \Phi$, $S_1 \subseteq S_2$; also, for each S, $\phi^1(S) \subseteq \phi^2(S)$ by the induction hypothesis. Therefore, $\nu\phi^1_{S_1} \cup S_1 \subseteq \nu\phi^2_{S_2} \cup S_2$.

□

If $H \vdash s \in \Phi$ has a successful tableau, and $H \subseteq_{\Phi} H'$, then this result allows us to conclude that $H' \vdash s \in \Phi$ also has a successful tableau without having to construct one. Likewise, if $H \vdash s \in \Phi$ does not have a successful tableau, and $H' \subseteq_{\Phi} H$, then it is also the case that $H' \vdash s \in \Phi$ does not have a successful tableau.

To conclude this section, we remark on a final property of tableaux. Given a sequent σ, define H_σ, s_σ and Φ_σ to be such that $\sigma \equiv H_\sigma \vdash s_\sigma \in \Phi_\sigma$. Suppose that $s : \nu X.\Phi \in H$ and that $H \vdash s' \in \nu X.\Phi$ has a successful tableau T involving a leaf $H' \vdash s \in \nu X.\Phi$ that is successful by virtue of the fact that $s : \nu X.\Phi \in H$ (in other words, $s : \nu X.\Phi \in H_{\sigma'}$ for all σ' between $H \vdash s' \in \nu X.\Phi$ and $H' \vdash s \in \nu X.\Phi$). In this case we say that T *depends on* $s : \nu X.\Phi$. Suppose further that $H - \{s : \nu X.\Phi\} \vdash s \in \nu X.\Phi$ is subsequently determined not to have a successful tableau. Then it is impossible to construct a successful tableau T' with root $H - \{s : \nu X.\Phi\} \vdash s' \in \nu X.\Phi$ that is built in the same way as T except that the sequents $H' \vdash s \in \nu X.\Phi$ that were leaves in T but are not in T' now have tableaux built for them. The import of this property is that information regarding the existence of successful tableaux for a sequent of the form $H \vdash s' \in \nu X.\Phi$ can be saved, even if some of the hypotheses of the form $s : \nu X.\Phi$ turn out to be false. In order to do so, every successful tableau for $H \vdash s' \in \nu X.\Phi$ must be computed, and the information regarding the hypotheses $s : \nu X.\Phi \in H$ that each depends on saved. If one of these hypotheses involving turns out not to be true, then the formerly successful tableau cannot be extended to a successful tableau for $H - \{s : \nu X.\Phi\} \vdash s' \in \nu X.\Phi$. If every successful tableau for $H \vdash s' \in \nu X.\Phi$ depends on $s : \nu X.\Phi$, then there can be no successful tableau for $H - \{s : \nu X.\Phi\} \vdash s' \in \nu X.\Phi$.

6 Conclusions

This paper has presented a tableau-based proof system for determining whether finite-state processes have properties expressible in the propositional mu-calculus and has laid some of the groundwork for an efficient model checking algorithm. Some of the strategies implied by the results in the previous section have been incorporated in an implementation of a model checker for the Concurrency Workbench [8, 9], a tool for the automated analysis of concurrent systems. Experience with the implementation has so far indicated that a reasonable degree of efficiency is possible. A variety of small examples involving processes having up to 300 states and formulas involving several nested fixed points have been tried, and on a Sun 4 workstation the response time has been in the range of 30 seconds.

The issue of efficiency needs a more thorough and systematic treatment. There is evidence to suggest that an $O((|S| * |\Phi|)^{id(\Phi)+1})$ algorithm can be derived on the basis of the results presented in the previous section, where $id(\Phi)$, the *interconnection depth* of Φ, is a measure of the degree of mutual recursion in Φ. Such a complexity result would be significant because a wide variety of logics may be encoded in the mu-calculus using formulas having interconnection depth ≤ 2; indeed, all the logics mentioned in [13] have this property. Thus, an algorithm of the stated complexity would yield cubic-time model checkers for these logics. It is also potentially the case that an $O((|S| * |\Phi|)^{ad(\Phi)+1})$ algorithm is possible, where $ad(\Phi)$ is the *alternation depth* of Φ, as defined in [13]. The significance of this result stems from the fact that this is the same worst-case complexity exhibited by the algorithm in [13]. Establishing such a result in our setting would require, among other things, a proof that the tableau system is sound and complete for sequents of the form $\emptyset \vdash s \in \Phi$ even when the relation $\prec$ in rules R7 and R8 is replaced by $\prec_{-}$. There appears to be a proof-theoretic argument to support this claim, although a complete proof has not yet been developed.

Another interesting question to pursue is the issue of compositionality alluded to in the introduction. Using techniques developed by Stirling [23, 24], it is likely that the proof system in this paper can be modified so that properties of concurrent systems can be deduced on the basis of the properties of their component processes. It could then be possible to develop a compositional model checker for the propositional mu-calculus which could then be used to check systems of parallel components efficiently.

Other researchers have considered the problem of model-checking in the propositional mu-calculus. Emerson and Lei develop an efficient semantics-based approach in [13]; using essentially the same semantic account as in figure 2 their algorithm computes the set of states corresponding to a formula in the logic. In order to determine if a state satisfies a proposition, then, one tests whether it is contained in the corresponding set. This approach can result in unnecessary information being computed, since it is often the case that it is not necessary to compute whether every state satisfies the formula in order to determine whether a particular one does; however, the asymptotic complexity of their algorithm is the best known. Arnold and Crubille [1] present a linear-time algorithm for solving recursive equations in a logic similar in spirit to the propositional mu-calculus; using their results, it is possible to develop a linear-time model checker for the fragment of our logic that does not contain any alternating fixed points. Independently of us, Stirling and Walker [25] and Winskel [28] have also pursued tableau-based approaches. The former essentially rename recursive propositions each time they are unwound to bound the number of unrollings, and their rules for recursive propositions contain, as a side-condition, a reference to the remainder of the tableau that has been constructed. These properties complicate their procedure somewhat, and it is unclear how they will be able to prove theorems that enable information to be reused; however, their proofs of soundness and completeness use very interesting proof-theoretic techniques. Winskel uses assumptions to bound the number of times recursive propositions are unrolled in a way similar to ours; however, he introduces these assumptions into the syntax of propositions, thereby adding notational complexity. Although he does not prove the kinds of results that we establish in section 5, his approach yields very clean proofs of soundness and completeness. In both cases, the decision procedures are of roughly the same complexity as *check1*. Larsen has also developed proof systems for fragments of the propositional mu-calculus, and they been implemented in a Prolog-based system [18]. His axiomatizations operate on sequents that are similar ours; however, his logics do not allow for propositions having both maximal and minimal fixed points and hence are not as expressive as the full mu-calculus, although they do allow a wide range of properties to be expressed.

Acknowledgements

I would like to thank Colin Stirling and David Walker for many interesting and stimulating discussions about the propositional mu-calculus, and to Colin especially for his technical advice and encouragement. Luca Aceto and Matthew Hennessy gave me patient support during the preparation of this paper, and Bernhard Steffen and Anna Ingólfsdóttir provided valuable comments on early versions of this paper.

References

[1] Arnold, A. and P. Crubille. "A Linear Algorithm To Solve Fixed-Point Equations on Transition Systems." *Information Processing Letters*, v. 29, 30 September 1988, pp. 57-66.

[2] Browne, M.C. "An Improved Algorithm for the Automatic Verification of Finite State Systems Using Temporal Logic." In *Proceedings of First Annual Symposium on Logic in Computer Science*, 1986, pp. 260-266.

[3] Browne, M.C., E.M. Clarke and D. Dill. "Checking the Correctness of Sequential Circuits." In *Proceedings of the 1985 IEEE International Conference on Computer Design*, pp. 545-548.

[4] Browne, M.C., E.M. Clarke and D. Dill. "Automatic Circuit Verification Using Temporal Logic: Two New Examples." In *Formal Aspects of VLSI Design*, G.J. Milne and P.A. Subrahmanyam eds., pp. 113-124. Elsevier-North Holland, New York, 1986.

[5] Browne, M.C., E.M. Clarke, D. Dill and B. Mishra. "Automatic Verification of Sequential Circuits Using Temporal Logic." *IEEE Transactions on Computing*, C-35(12), pp. 1035-1044.

[6] Clarke, E.M, D.E. Long and K.L. McMillan. "Compositional Model Checking." In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science*, 1989. Computer Society Press, Washington DC.

[7] Clarke, E.M., E.A. Emerson and A.P. Sistla. "Automatic Verification of Finite State Concurrent Systems Using Temporal Logic Specifications." *ACM Transactions on Programming Languages and Systems*, v. 8, n. 2, 1986, pp. 244-263.

[8] Cleaveland, W.R., J. Parrow and B.U. Steffen. "A Semantics-Based Tool for the Verification of Finite-State Systems." In *Proceedings of the Ninth IFIP Symposium on Protocol Specification, Testing and Verification*, June 1989. To be published by North-Holland.

[9] Cleaveland, W.R., J. Parrow and B.U. Steffen. "The Concurrency Workbench." In *Proceedings of the Workshop on Automatic Verification Methods for Finite-State Systems.* To be published in the Lecture Notes in Computer Science series, Springer-Verlag, Berlin.

[10] Emerson, E.A. "Branching Time Temporal Logic: A Tutorial." In *Proceedings of the REX Summer School/Workshop*, Noordwijkerhout, the Netherlands, 1988.

[11] Emerson, E.A. and E.M. Clarke. "Characterizing Correctness Properties of Parallel Programs as Fixpoints." In *Proceedings of the Seventh International Colloquium on Automata, Languages and Programming*, Lecture Notes in Computer Science 85. Springer-Verlag, Berlin, 1981.

[12] Emerson, E.A. and J.Y. Halpern. "'Sometimes' and 'Not Never' Revisited: On Branching versus Linear Time." In *Proceedings the the Twelfth Annual ACM Symposium on Principles of Programming Languages*, 1983.

[13] Emerson, E.A. and C.-L. Lei. "Efficient Model Checking in Fragments of the Propositional Mu-Calculus." In *Proceedings of the First Annual Symposium on Logic in Computer Science*, 1986, pp. 267-278.

[14] Fischer, M.J. and R.E. Ladner. "Propositional Dynamic Logic of Regular Programs." *Journal of Computing and System Sciences*, v. 18, 1979, pp. 194-211.

[15] Hennessy, M. and R. Milner. "Algebraic Laws for Nondeterminism and Concurrency." *Journal of the Association for Computing Machinery*, v. 32, n. 1, January 1985, pp. 137-161.

[16] Kozen, D. "Results on the Propositional μ-Calculus." *Theoretical Computer Science*, v. 27, 1983, pp. 333-354.

[17] Lamport, L. "'Sometimes' is Sometimes 'Not Never'–On the Temporal Logic of Programs." In *Proceedings of the Seventh Annual ACM Symposium on Principles of Programming Languages*, 1980, pp. 174-185.

[18] Larsen, K.G. "Proof Systems for Hennessy-Milner Logic with Recursion." In *Proceedings of CAAP*, 1988.

[19] Milner, R. *A Calculus of Communicating Systems.* Lecture Notes in Computer Science 92. Springer-Verlag, Berlin, 1980.

[20] Parrow, J. "Submodule Construction as Equation Solving in CCS." In *Proceedings of the Foundations of Software Technology and Theoretical Computer Science*, Lecture Notes in Computer Science 287, pp. 103-123. Springer-Verlag, Berlin, 1987.

[21] Pnueli, A. "Linear and Branching Structures in the Semantics and Logics of Reactive Systems." In *Proceeding of the Twelfth International Conference on Automata, Languages and Programming*, Lecture Notes in Computer Science 194, pp. 14-32. Springer-Verlag, Berlin, 1985.

[22] Steffen, B.U. "Characteristic Formulae for CCS with Divergence." In *Proceedings of Eleventh International Colloquium on Automata, Languages and Programming*, 1989.

[23] Stirling, C. "A Complete Modal Proof System for a Subset of SCCS." In *Proceedings of TAPSOFT 85*, Lecture Notes in Computer Science 185, pp. 253-266. Springer-Verlag, Berlin, 1985.

[24] Stirling, C. "Modal Logics for Communicating Systems." *Theoretical Computer Science*, v. 49, 1987, pp. 311-347.

[25] Stirling, C. and D. Walker. "Local Model Checking in the Modal Mu-Calculus." In *Proceedings of TAPSOFT '89*, Lecture Notes in Computer Science 351. Springer-Verlag, Berlin, 1989.

[26] Tarski, A. "A Lattice-Theoretical Fixpoint Theorem and its Applications." *Pacific Journal of Mathematics*, v. 5, 1955.

[27] Vardi, M.Y. and P. Wolper. "An Automata-Theoretic Approach to Automatic Program Verification." In *Proceedings of the First Annual Symposium on Logic in Computer Science*, 1986, pp. 332-344.

[28] Winskel, G. "Model Checking in the Modal ν-Calculus." In *Proceedings of Eleventh International Colloquium on Automata, Languages and Programming*, 1989.