Prefixed Tableaus and Nested Sequents[*]

Melvin Fitting
Dept. Mathematics and Computer Science
Lehman College (CUNY), 250 Bedford Park Boulevard West
Bronx, NY 10468-1589
e-mail: melvin.fitting@lehman.cuny.edu
web page: comet.lehman.cuny.edu/fitting

September 14, 2011

Abstract

Nested sequent systems for modal logics are a relatively recent development, within the general area known as deep reasoning. The idea of deep reasoning is to create systems within which one operates at lower levels in formulas than just those involving the main connective or operator. Prefixed tableaus go back to 1972, and are modal tableau systems with extra machinery to represent accessibility in a purely syntactic way. We show that modal nested sequents and prefixed modal tableaus are notational variants of each other, roughly in the same way that Gentzen sequent calculi and tableaus are notational variants. This immediately gives rise to new modal nested sequent systems which may be of independent interest. We discuss some of these, including those for some justification logics that include standard modal operators.

1   Introduction

Nested sequent systems have been introduced for a variety of logics. These can be thought of as sequent systems, but in which rules can be applied well within a formula, and not just to the top level connective, and thus they use a bit of deep inference. The mechanism for this is, as the name suggests, sequents within sequents within etc. Of the nested sequent systems that have been considered, here we are interested in the modal ones, [4, 5]. See `http://alessio.guglielmi.name/res/cos/index.html` for general information about the current range of work on deep inference.

It is well-known that the classical semantic tableau calculus and the classical Gentzen sequent calculus are essentially the same thing—one is the other upside down. This is most easily seen if *signed* tableaus are used; each formula has either T or F as a sign. Then a tableau branch converts to a sequent by putting on the left of the sequent arrow all formulas that appear on the branch with a sign of T, and on the right of the arrow all those with a sign of F. The sequence of tableau steps needed to produce a closed tableau for $F\,X$, reversed, becomes the sequence of sequent steps needed to produce a Gentzen system proof of $\rightarrow X$. Indeed in [24], tableaus and sequents were sometimes treated simultaneously as instances of a single more abstract notion. Further, what applies to classical logic also applies to tableau/sequent calculi for modal logics, intuitionistic logic, and so on. One must exercise some care here, however. Sequent calculi are often formulated using contraction and weakening rules, which are uncommon in a tableau context. However, if sequents

are understood to involve sets of formulas, instead of sequences or multisets of formulas, contraction and weakening are unnecessary. If a sequent calculus is formulated this way, then for each sequent calculus for a logic it is relatively straightforward to produce a corresponding tableau calculus, and conversely. Indeed, even contraction and weakening can be brought into the tableau formulation, to produce tableaus for substructural logics. One might naturally say, then, that sequent calculi and tableau systems are notational variants of each other.

In 1972 *prefixed* tableaus were introduced for modal logics, [10], and presented at greater length in [11]. These add extra machinery, prefixes, to the usual tableau methodology, making it possible to give tableau proof systems for several modal logics that lack those of the ordinary kind. In [18] a number of the original rules were replaced by new ones that are more modular, and the resulting systems have become widely adopted. See [13, 15] for a thorough discussion.

In this paper we show that prefixed modal tableau systems and modal nested sequent systems are essentially the same thing, in the same way that classical tableau proofs and classical sequent proofs are. That there is a relationship of this sort was mentioned in [5], "The tableau formalism which corresponds most closely to nested sequents is the prefixed tableau formalism... In particular, prefixes impose the same tree structure on formulas that is imposed in a nested sequent," but this observation was not followed up on. We do so in this paper. Since prefixed tableaus have had a substantial development history, one consequence of the present paper is that a number of nested sequent systems of possible interest immediately become available.

When translating tableaus to Gentzen sequents *signed* systems are most useful. But nested sequent systems are Tait style, one-sided, and for these systems *unsigned* tableaus relate better and are what we will use. We need a translation procedure from prefixed tableaus to the kind of machinery used with nested sequents, analogous to the simple one described above: 'put the T formulas on the left of the arrow and the F formulas on the right.' It turns out that the translation was given in essence long ago, in [11]. The translation was originally used as one way of proving completeness of an axiomatic modal logic with the Barcan formula.

The plan of this paper is as follows. We first sketch a prefixed modal tableau system, followed by a presentation of the formalism of modal nested sequents. We then give our translation procedure. It is quite transparent, so we do not give a formal proof of correctness. We do, however, follow an example through in detail. Since tableau completeness proofs can be given rather easily, completeness of corresponding nested sequent systems follows directly. Tableau systems can be proved complete either by giving a systematic tableau construction algorithm, as in [11], or by using maximal consistent set constructions, as in [13]—an example of the maximal set approach is presented in Section 12. Our discussion is not intended to provide any substitutes for the important proof theoretical results that have been established using nested sequents. Finally, Kurokawa has created prefixed tableau systems for some Justification Logics, in [17]. We conclude with these, and their nested sequent counterparts, to illustrate how work in one domain induces results in the other.

I want to thank Roman Kuznets for his close reading of earlier versions of this paper, for his suggestions, and for insisting I could do better. I also want to thank an anonymous referee for extensive and detailed comments that substantially improved the presentation in this paper.

2   Prefixed Tableaus

Nested sequent systems typically work with formulas in negation normal form, with $\wedge$ and $\vee$ as basic connectives, but there is no particular advantage to such a restriction for tableau systems, and we will not impose it on either tableaus or nested sequents, though it could easily be brought

in. For us, formulas are built up from propositional letters, P, Q, $\ldots$, using $\wedge$, $\vee$, $\neg$, $\supset$, $\Box$, and $\Diamond$ in the usual way.

Formulas may be of primary interest, but the basic building block of prefixed tableau proofs is the *prefixed* formula, something that can be traced back to [7]. Intuitively, a prefix is a name for a possible world. We want a system of prefixes that provides a purely syntactic and structural representation of the accessibility relation—this is where prefixed tableaus differ from the more general labeled systems of [21], where accessibility is expressed directly as a relational formula. A *prefix* is a non-empty finite sequence of positive integers beginning with 1, such as 1.3.2.1.4 (we use periods as separators). Think of the actual world as 1. If n is a positive integer and σ is a prefix, by $\sigma.n$ we mean the result of adjoining n to the end of σ. Think of $\sigma.n$ as naming a world that is accessible from the world that is named by σ. A *prefixed formula* is of the form $\sigma\, X$ where σ is a prefix and X is a formula. Think of $\sigma\, X$ as saying X is true at the world named by σ.

It is convenient to group formulas into classes that behave alike—an α, β, ν, π classification is standard and we use it here. It is often referred to as *uniform notation*. First, compound propositional formulas and their negations are grouped into those that behave conjunctively, the α formulas, and those that behave disjunctively, the β formulas. For each, *components* are defined, α_1 and α_2 for α formulas, and β_1 and β_2 for β formulas. These are given in the following tables.

α	α_1	α_2		β	β_1	β_2
$X \wedge Y$	X	Y		$X \vee Y$	X	Y
$\neg(X \vee Y)$	$\neg X$	$\neg Y$		$\neg(X \wedge Y)$	$\neg X$	$\neg Y$
$\neg(X \supset Y)$	X	$\neg Y$		$X \supset Y$	$\neg X$	Y

In each case an α formula is true just in case both α_1 and α_2 are true; a β formula is true if one of β_1 or β_2 is. The tables can be extended to include more binary connectives, but these standard ones will suffice here.

In a similar way we define ν, necessary, and π, possible, formulas and negated formulas, along with their components, in the following tables. Here the idea is, a ν formula is true at a possible world of a Kripke model just in case ν_0 is true at every accessible world; a π formula is true at a possible world if π_0 is true at some accessible world.

ν	ν_0		π	π_0
$\Box X$	X		$\Diamond X$	X
$\neg \Diamond X$	$\neg X$		$\neg \Box X$	$\neg X$

A prefixed tableau is a tree meeting certain conditions, in which each node is labeled with a prefixed formula. We give rules for starting, continuing, and terminating a tableau construction. The rules are for K, the simplest normal modal logic, and go back to [10]. There are similar rules for other standard modal logics, presented in Section 7. We give the rules for constructing a prefixed tableau for a finite set S of prefixed formulas. One can think of the construction of such a tableau as a search for information about a model for S. We also define what it means for a tableau to be closed—intuitively, a closed tableau for S establishes that S has no models. This will be made more precise below.

Let S be a finite set of prefixed formulas. A one-branch tree with the members of S as node labels is a *tableau for S*. (The order in which members of S appear does not matter.)

Next, if $\mathcal{T}$ is a tableau for S, and a *branch extension rule* is applied to $\mathcal{T}$ to turn it into $\mathcal{T}'$, then $\mathcal{T}'$ is also a tableau for S. We now state the branch extension rules for K.

The classical branch extension rules are as follows.

¬¬ **Rule** A branch containing $\sigma \, \neg\neg X$ may be extended with a node labeled $\sigma \, X$.

α **Rule** A branch containing $\sigma \, \alpha$ may be extended with two nodes labeled $\sigma \, \alpha_1$ and $\sigma \, \alpha_2$.

β **Rule** A branch containing $\sigma \, \beta$ may be split, with each fork extended by a single node, one labeled $\sigma \, \beta_1$, the other $\sigma \, \beta_2$.

Schematically the classical branch extension rules look like the following.

Prefix Tableau Double Negation Rule	$$\dfrac{\sigma \, \neg\neg X}{\sigma \, X}$$
Prefix Tableau α Rule	$$\dfrac{\sigma \, \alpha}{\substack{\sigma \, \alpha_1 \\ \sigma \, \alpha_2}}$$
Prefix Tableau β Rule	$$\dfrac{\sigma \, \beta}{\sigma \, \beta_1 \mid \sigma \, \beta_2}$$

Next we give the branch extension rules for the modal cases. We say a prefix σ *occurs on a branch* if there is a prefixed formula $\sigma' \, X$ on the branch, and σ is an initial segment of σ', not necessarily proper. Thus, if $1.2.1.3 \, X$ is on a branch, 1.2 is understood to occur, for example.

ν **Rule** A branch containing $\sigma \, \nu$ may be extended with $\sigma.n \, \nu_0$ provided the prefix $\sigma.n$ already occurs on the branch.

π **Rule** A branch containing $\sigma \, \pi$ may be extended with $\sigma.n \, \pi_0$ provided the prefix $\sigma.n$ is new to the branch.

Schematically, these rules are as follows.

Prefix Tableau ν Rule	$$\dfrac{\sigma \, \nu}{\substack{\sigma.n \, \nu_0 \\ \sigma.n \text{ not new}}}$$
Prefix Tableau π Rule	$$\dfrac{\sigma \, \pi}{\substack{\sigma.n \, \pi_0 \\ \sigma.n \text{ new}}}$$

Intuitive motivation is straightforward. If $\Box X$ is true at the world named by σ, then X is true at any accessible world, in particular at the one named by $\sigma.n$, which has been established to exist because $\sigma.n$ already occurs on the branch. Hence the $\Box$ Rule, one of the two cases incorporated in the ν Rule. If $\Diamond X$ is true at the world named by σ then X must be true at some accessible world. We can give this world a name, $\sigma.n$, but the name must be uncommitted, hence the newness condition in the π Rule.

A *tableau branch* is *closed* if it contains both $\sigma \, X$ and $\sigma \, \neg X$ for some formula X. A *tableau* is *closed* if each branch is closed.

Finally, we relate this to formulas as such. A prefix tableau proof of a formula X is a closed tableau for the set $\{1 \, \neg X\}$. The intuitive idea is that to prove X we suppose there is some possible
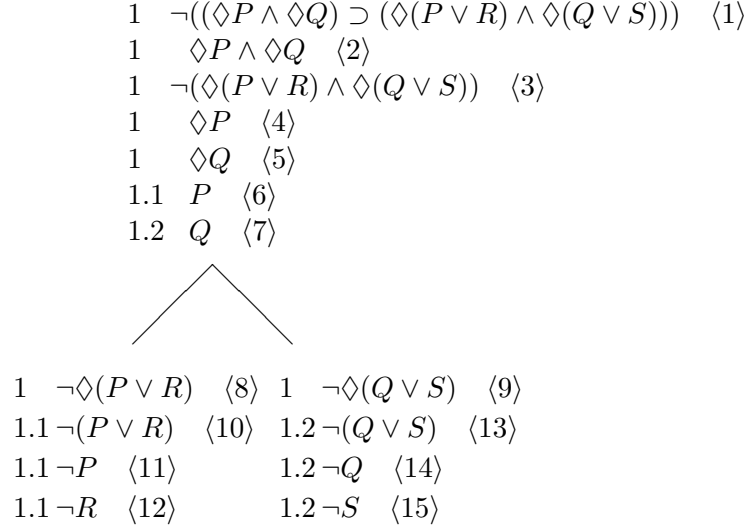
$$1 \quad \neg((\Diamond P \wedge \Diamond Q) \supset (\Diamond(P \vee R) \wedge \Diamond(Q \vee S))) \quad \langle 1 \rangle$$
$$1 \quad \Diamond P \wedge \Diamond Q \quad \langle 2 \rangle$$
$$1 \quad \neg(\Diamond(P \vee R) \wedge \Diamond(Q \vee S)) \quad \langle 3 \rangle$$
$$1 \quad \Diamond P \quad \langle 4 \rangle$$
$$1 \quad \Diamond Q \quad \langle 5 \rangle$$
$$1.1 \quad P \quad \langle 6 \rangle$$
$$1.2 \quad Q \quad \langle 7 \rangle$$

$1 \quad \neg\Diamond(P \vee R) \quad \langle 8 \rangle$	$1 \quad \neg\Diamond(Q \vee S) \quad \langle 9 \rangle$
$1.1 \, \neg(P \vee R) \quad \langle 10 \rangle$	$1.2 \, \neg(Q \vee S) \quad \langle 13 \rangle$
$1.1 \, \neg P \quad \langle 11 \rangle$	$1.2 \, \neg Q \quad \langle 14 \rangle$
$1.1 \, \neg R \quad \langle 12 \rangle$	$1.2 \, \neg S \quad \langle 15 \rangle$

Figure 1: K Tableau Proof Example

world, call it 1, where X fails, we derive a contradiction, and we conclude X must then be the case at arbitrary worlds.

Figure 1 displays an example of a K closed tableau. It is a proof for the formula $(\Diamond P \wedge \Diamond Q) \supset (\Diamond(P \vee R) \wedge \Diamond(Q \vee S))$. We will build on this proof in later examples. We must construct a closed tableau for the set $\{1 \, \neg((\Diamond P \wedge \Diamond Q) \supset (\Diamond(P \vee R) \wedge \Diamond(Q \vee S)))\}$. This set has only one member, and this gives us line $\langle 1 \rangle$. Lines $\langle 2 \rangle$ and $\langle 3 \rangle$ are from $\langle 1 \rangle$ by α, as are $\langle 4 \rangle$ and $\langle 5 \rangle$ from $\langle 2 \rangle$. Line $\langle 6 \rangle$ is from $\langle 4 \rangle$ by π as is $\langle 7 \rangle$ from $\langle 5 \rangle$; note that 1.1 and 1.2 are new prefixes at this point. Lines $\langle 8 \rangle$ and $\langle 9 \rangle$ are from $\langle 3 \rangle$ by β. Line $\langle 10 \rangle$ is from $\langle 8 \rangle$ by ν; note that 1.1 already occurs on the branch at this point. Lines $\langle 11 \rangle$ and $\langle 12 \rangle$ are from $\langle 10 \rangle$ by α. The left branch is closed by $\langle 6 \rangle$ and $\langle 11 \rangle$. The right branch is similar.

Completeness can be proved by giving a systematic tableau construction procedure. This leads to an easy decision procedure—such a procedure is described in detail in [11]. Alternatively one can use a maximal consistent set construction, which we do in Section 12 for a different, but related, logic.

There are two restrictions that can be placed on tableaus, both of which play a role in relating them to nested sequents. We discuss them now.

First, we have called a tableau closed if each branch contains a syntactic contradiction, σX and $\sigma \neg X$. Let us say a tableau is *atomically* closed if each branch contains σP and $\sigma \neg P$, where P is some *propositional letter*. Restricting tableau proofs so that atomic closure is required does not change the class of provable formulas.

Second, many of the tableau rules can be restricted to be *single-use*. The single-use restriction says that such-and-such a rule can be applied to a particular prefixed formula at most once on any given branch. In fact, all the tableau rules stated so far, except for the ν Rule, can have single-use restrictions placed on them, without changing the class of provable formulas. If the only rule that applies to a given prefixed formula in a tableau is a single-use rule, the formula can be 'crossed off' after the rule is applied, and this simplifies the search space somewhat.

In Section 12 we give a sample tableau completeness proof, using a maximal consistent set construction. That construction actually proves completeness even with atomic closure and single-

use restrictions imposed, as we will discuss at that point. Note that the example shown in Figure 1 meets these restrictions. (As it happens, in this example all rule applications are single-use, but in general such a restriction cannot be imposed on the ν rule without compromising completeness.)

3   Nested Sequent Systems

We sketch a variant of the modal nested sequent system from [4], just for the logic K for now. We have made some changes to the original formulation, to fit better with prefixed tableaus as we have just presented them. First, we do not assume formulas are in negation normal form; we allow arbitrary use of $\land$, $\lor$, $\supset$, and $\neg$. Second, [4] works with multisets, and includes contraction and weakening rules. We are not interested in proof-theoretic issues here, so it will be simpler to use sets in place of multisets, and drop all structural rules. Third, we make use of uniform notation, α, β, ν, and π.

Gentzen-style sequents, of the form $S_1 \to S_2$, are familiar things. One thinks of $S_1 \to S_2$ as asserting that the conjunction of the formulas in S_1 implies the disjunction of the formulas in S_2. Somewhat less familiar are Tait-style one-sided sequents, in which an arrow does not appear. Think of a one-sided sequent as a variant of a Gentzen sequent, but in which all formulas have been moved to the right of the arrow (thus negating them), and with the arrow no longer written. Then a one-sided sequent is just a set of formulas that should be thought of as a generalized disjunction, since this is how right sides of arrows are generally understood. (Recall, we are using sets here, rather than multisets.) Nested sequents for modal logic simply iterate this idea. Loosely speaking, nesting corresponds to necessitation.

Definition 3.1 A *nested sequent* is a non-empty finite set of formulas and nested sequents.

Note: empty nested sequents could be allowed, as they are in [4, 5]. The connection with prefixed tableaus is simpler if we do not do so, and so they are ruled out here.

Ordinary Tait-style sequents are thought of as disjunctions. This extends to nested sequents via the following translation. Let $\Gamma = \{X_1, \ldots, X_n, \Delta_1, \ldots, \Delta_k\}$ be a nested sequent, where each X_i is a formula and each Δ_j is a nested sequent. Then

$$\Gamma^\dagger = X_1 \lor \ldots \lor X_n \lor \Box\Delta_1^\dagger \lor \ldots \lor \Box\Delta_k^\dagger$$

For each nested sequent Γ, one can think of $\Gamma^\dagger$ as its 'meaning' expressed as a formula.

Certain conventions have become standard for writing nested sequents, and we generally follow them here. First, while we are working with sets, all enclosing curly brackets are omitted. Second, a nested sequent that is a member of another nested sequent is represented by listing its members in square brackets, and is called a *boxed sequent*. As above, we use Γ, Δ, ... for nested sequents.

Example $\Gamma = \{A, B, \{C, \{D, E\}, \{F, G\}\}\}$ is a nested sequent (where the letters stand for formulas). We will generally write Γ as: $A, B, [C, [D, E], [F, G]]$. For this, $\Gamma^\dagger = A \lor B \lor \Box(C \lor \Box(D \lor E) \lor \Box(F \lor G))$.

When working with nested sequents one commonly defines notions of holes, multiple holes, and contexts—see [4] for a full definition. We will not need this machinery for present purposes, and so we replace it with the following simpler items.

Definition 3.2 We define a notion of *subsequent* as follows.

1. Nested sequent Γ is a subsequent of Γ.

2. If Γ and Δ are nested sequents and $\Delta \in \Gamma$, any subsequent of Δ is a subsequent of Γ.

We say a formula X *occurs in* nested sequent Γ if $X \in \Delta$ for some subsequent Δ of Γ. We say X occurs *directly* in Γ if $X \in \Gamma$.

In the example above, Γ has four subsequents: itself, $\{C, \{D, E\}, \{F, G\}\}$, $\{D, E\}$, and $\{F, G\}$. All of the formulas A, B, C, D, E, F, and G occur in Γ; A and B occur directly.

Suppose Γ is a nested sequent in which the propositional letter P occurs, but only once. We write $\Gamma(P)$ to indicate this. Later if we write $\Gamma(X)$ we mean the result of replacing the formula P in Γ with the formula X. Likewise we write $\Gamma(X, Y)$ to mean the result of replacing P in Γ with the two formulas X and Y. Similarly $\Gamma(\Delta)$ means the result of replacing P with the nested sequent Δ. And so on. For instance, if $\Gamma(P)$ is the nested sequent $A, [B, [C, P]]$ and Δ is $D, E, [F]$ then $\Gamma(\Delta)$ is the nested sequent $A, [B, [C, D, E, [F]]]$. Note that Δ might not be a subsequent of $\Gamma(\Delta)$—it is not in the example just given.

We now state the nested sequent rules for K, from [4], extended to allow arbitrary formulas and not just those in negation normal form. Rules can be given for other standard modal logics, but we postpone them until Section 7. Assume $\Gamma(P)$ is any nested sequent with one occurrence of propositional letter P. Also, when we write $[\ldots]$ we refer to a *non-empty* nested sequent, and $[Z, \ldots]$ is the same sequent but with Z added. This notation is used in the π rule here, and in various rules later on as well.

Nested Sequent Axioms $\qquad\qquad\qquad \Gamma(A, \neg A),$

$$A \text{ a propositional letter}$$

Nested Sequent Double Negation Rule $\qquad \dfrac{\Gamma(X)}{\Gamma(\neg\neg X)}$

Nested Sequent α Rule $\qquad \dfrac{\Gamma(\alpha_1) \qquad \Gamma(\alpha_2)}{\Gamma(\alpha)}$

Nested Sequent β Rule $\qquad \dfrac{\Gamma(\beta_1, \beta_2)}{\Gamma(\beta)}$

Nested Sequent ν Rule $\qquad \dfrac{\Gamma([\nu_0])}{\Gamma(\nu)}$

Nested Sequent π Rule $\qquad \dfrac{\Gamma(\pi, [\pi_0, \ldots])}{\Gamma(\pi, [\ldots])}$

Sequent proofs start with axioms and end with the nested sequent being proved. The fundamental notion is that of proof of a sequent, with proof of a formula as a derivative notion. A proof of the nested sequent consisting of just the formula X is understood to be a proof of X itself. An example of a proof is given in Figure 3.

Soundness and completeness are proved in [4] for a family of modal nested sequent calculi, but the formulation here is somewhat different so it is not immediate that Brünnler's results carry over to the present version. As originally formulated formulas were in negation normal form, a condition we have relaxed. Also empty sequents were allowed, which is not the case here. Nonetheless, soundness and completeness hold. We will show how to translate back and forth between our nested sequent systems and prefixed tableau systems, and this transfers to nested sequents the soundness and completeness results for prefixed tableau systems, which can be found in many

places in the literature. But in fact, Brünnler's original arguments can be carried over directly, as we now very briefly sketch.

For soundness one shows by induction on proof length that, if Γ is a line in a nested sequent proof then $\Gamma^\dagger$ is a formula valid in all Kripke models (for the logic K, which is all we have considered so far). This is shown for axioms by induction on formula complexity. For the rules, one shows that the conjunction of translates of rule premises implies the translate of the conclusion. Of course this requires several cases, together with some results about substitution, but it is all rather direct. Note that if a proof ends with the formula X, we have that $\{X\}^\dagger = X$, and so the system only proves valid formulas.

Of course completeness is more complicated to establish, though it is simpler for K than for some other modal logics. Essentially, for a formula X one constructs a sequent proof attempt bottom up, systematically applying rules backwards. For K this will either terminate with an actual proof, or with an incomplete proof in which everything that could be done has been. From such a completely constructed but failed proof a counter-model can be constructed. Take an upward 'branch' in the failed proof that does not terminate with an axiom at the top, and use the top sequent to construct a model. Take its nested sequents to be possible worlds, with nesting corresponding to accessibility, and take atomic membership as determining a valuation in the model. Then a so-called 'Truth Lemma' can be shown. We omit details here—they are similar to one of the ways completeness is proved for prefixed tableaus.

In addition, one can give a constructive, syntactic, proof of cut elimination for the nested sequent system, and then derive completeness from that of an axiomatic system for K. A direct cut elimination argument can be found in [4].

4 Turning Tableaus Over

A tableau system is a backward reasoning system; by contrast a nested sequent system is forward reasoning. As a preliminary part of a translation procedure, we show how to *invert* prefixed tableaus into forward reasoning versions, but with prefixes still present. Tableaus start with a finite set S of prefixed formulas and terminate with contradictions. An inverted system would start with universal truths and conclude with $\overline{S}$, which is like S except that formulas have been negated. Throughout, individual formulas need to be negated, and the order of rule applications must be inverted. Both aspects need some discussion.

Negating formulas, while conceptually simple, raises some syntactical difficulties. Clearly negating $X \wedge Y$ is unproblematic—we get $\neg(X \wedge Y)$. But what about negating $\neg X$; should we get X or $\neg\neg X$? And what about negating $\neg\neg X$, and so on? Any particular choice carries some difficulties with it, and this ultimately comes from our decision to allow arbitrary formulas in proofs, and not just those in negation normal form. Nonetheless, things are not too bad. We adopt the following. If formula X does not begin with a negation, the *inverse* of X is $\neg X$. For a formula, $\neg X$, beginning with a negation, the *inverse* is X. We write the inverse of formula Z as $\overline{Z}$, and for prefixed formulas we take $\overline{\sigma\, Z}$ to be $\sigma\, \overline{Z}$.

Next tableau rules must be turned over, with the formulas involved inverted. We can think of the set of prefixed formulas on a tableau branch as conjunctively connected. Once proofs are turned over, with formulas inverted, conjunctions become disjunctions. We begin discussion with the simplest case, the Prefix Tableau Double Negation Rule. Suppose we have a tableau branch on which the set of prefixed formulas is $S \cup \{\sigma\, \neg\neg X\}$, with $\sigma\, \neg\neg X$ not in S. We write this more simply as $S, \sigma\, \neg\neg X$. The double negation tableau rule allows us to extend the branch with $\sigma\, X$. But recall, double negation is a single-use rule, as discussed at the end of Section 2, and it is the

only rule that applies to $\sigma \,\neg\neg X$. This means we need never work with $\sigma \,\neg\neg X$ again on this branch, so without any loss we may think of it as having been removed. Then the conclusion of the tableau rule application, instead of being $S, \sigma \,\neg\neg X, \sigma \, X$, can just be taken to be $S, \sigma \, X$. Thus the tableau double negation rule can be stated as follows.

$$\frac{S, \sigma \,\neg\neg X}{S, \sigma \, X} \tag{1}$$

Turning rule (1) over and inverting formulas gives us the following, writing $\overline{S}$ for the inverses of the members of S and with sets now thought of disjunctively.

$$\frac{\overline{S}, \sigma \, \overline{X}}{\overline{S}, \sigma \, \overline{\neg\neg X}} \tag{2}$$

Using our definition of formula inverse, if X does not begin with a negation, rule (2) says the following.

$$\frac{\overline{S}, \sigma \,\neg X}{\overline{S}, \sigma \,\neg X} \tag{3}$$

Obviously rule (3) is useless, and need not be stated. But, if X does begin with a negation, say it is $\neg Y$, then rule (2) says the following, which is non-trivial.

$$\frac{\overline{S}, \sigma \, Y}{\overline{S}, \sigma \,\neg\neg Y} \tag{4}$$

Rule (4) will be formally adopted.

Next let us consider an instance of the Prefix Tableau β Rule, say where β is $X \vee Y$. Once again we have a single-use rule, so it can be stated as follows.

$$\frac{S, \sigma \, X \vee Y}{S, \sigma \, X \mid S, \sigma \, Y} \tag{5}$$

Inverting this, we get the following forward reasoning rule.

$$\frac{\overline{S}, \sigma \, \overline{X} \qquad \overline{S}, \sigma \, \overline{Y}}{\overline{S}, \sigma \, \overline{X \vee Y}} \tag{6}$$

We now have four cases to consider, instead of the two that we had with double negation. Suppose first that in (6) neither X nor Y begin with negations. Then (6) becomes the following, introducing an α formula, with α_1 and α_2 as premises.

$$\frac{\overline{S}, \sigma \,\neg X \qquad \overline{S}, \sigma \,\neg Y}{\overline{S}, \sigma \,\neg(X \vee Y)} \tag{7}$$

As another possibility, suppose that in (6) both X and Y begin with negations, say $X = \neg U$ and $Y = \neg V$. Then (6) becomes the following.

$$\frac{\overline{S}, \sigma \, U \qquad \overline{S}, \sigma \, V}{\overline{S}, \sigma \,\neg(\neg U \vee \neg V)} \tag{8}$$

This is awkward as it stands, because the conclusion involves an α formula, but the premises do not involve α_1 and α_2. For these to be brought in, the form of the rule should appear as follows.

$$\frac{\overline{S}, \sigma \,\neg\neg U \qquad \overline{S}, \sigma \,\neg\neg V}{\overline{S}, \sigma \,\neg(\neg U \vee \neg V)} \tag{9}$$

Fortunately, if we adopt (9) as a rule version then (8) becomes a derived rule if we make use of double negation introduction, (4), and this is the course we take. The other two cases of (6) are similar to the two we discussed and we leave them to the reader. In each case we can formulate things so that an α is derived from α_1 and α_2, possibly with some added double negation introductions. The tableau β rule, then, converts to a prefix sequent α rule.

Similar analyses apply to inversions of all the tableau rules of Section 2. The ν rule, which inverts to a π rule, requires some additional care, however, since it is not a single-use rule. The effects of this can be seen in the extra premise for the Prefix Sequent π Rule, as stated below.

Axioms could be taken to be sequents of the form $S, \sigma\,X, \sigma\,\neg X$, but in accordance with the remarks at the end of Section 2, X can be restricted to be a propositional letter.

The final result of all this is that the tableau rules from Section 2 invert to give us the following forward reasoning system for K, in which S is a set of prefixed formulas, possibly empty.

Prefix Sequent Axioms	$S, \sigma\,A, \sigma\,\neg A,$
	A a propositional letter
Prefix Sequent Double Negation Rule	$\dfrac{S, \sigma\,X}{S, \sigma\,\neg\neg X}$
Prefix Sequent α Rule	$\dfrac{S, \sigma\,\alpha_1 \qquad S, \sigma\,\alpha_2}{S, \sigma\,\alpha}$
Prefix Sequent β Rule	$\dfrac{S, \sigma\,\beta_1, \sigma\,\beta_2}{S, \sigma\,\beta}$
Prefix Sequent ν Rule	$\dfrac{S, \sigma.n\,\nu_0}{S, \sigma\,\nu}$
Prefix Sequent π Rule	$\dfrac{S, \sigma\,\pi, \sigma.n\,\pi_0}{S, \sigma\,\pi}$

In the ν rule above, the prefix $\sigma.n$ must not occur in the consequent, that is, in the sequent below the line. This amounts to requiring that $\sigma.n$ must not occur in S. In the π rule, the prefix $\sigma.n$ must occur in S. It is understood that $\sigma.n$ occurs in S if there is a member, $\sigma' Z$ of S with $\sigma.n$ as an initial segment of σ', not necessarily proper.

A prefix sequent proof ending with a finite set S of prefixed formulas is taken to be a proof of S. In particular, we understand a prefix sequent proof of $1\,X$ to be a proof of the formula X itself.

The tableau proof given in Figure 1 converts into a forward reasoning proof in the prefix sequent system, displayed in Figure 2. To produce it we have replaced tableau branches with prefix sequents, inverted the order of tableau rule applications, negated unnegated formulas. and unnegated negated ones. As it happens, it was not necessary to add any double negation introduction rule applications, though this is not the case with every example. The result is a proof in the prefix sequent system for K. We have included reference numbers, in angle brackets to set them off. These are intended to match corresponding steps in the tableau from Figure 1.

In the forward derivation of Figure 2, $\langle 11, 12 \rangle$ is a prefix sequent axiom because it contains $1.1\,\neg P$ and $1.1\,P$. The application of a β rule to it produces $\langle 10 \rangle$. In the original tableau, Figure 1,

$$\cfrac{\cfrac{1.1\,\neg P, 1.2\,\neg Q, 1\,\Diamond(P\vee R),}{1.1\,P, 1.1\,R \quad \langle 11, 12\rangle}}{\cfrac{1.1\,\neg P, 1.2\,\neg Q, 1\,\Diamond(P\vee R),}{1.1\,P\vee R, \quad \langle 10\rangle}}$$

1.1 ¬P, 1.2 ¬Q, 1 ◊(P ∨ R),
1.1 P, 1.1 R   ⟨11, 12⟩
_____
1.1 ¬P, 1.2 ¬Q, 1 ◊(P ∨ R),
1.1 P ∨ R,   ⟨10⟩
_____
1.1 ¬P, 1.2 ¬Q, 1 ◊(P ∨ R)   ⟨8⟩

1.1 ¬P, 1.2 ¬Q, 1 ◊(Q ∨ S),
1.2 Q, 1.2 S   ⟨14, 15⟩
_____
1.1 ¬P, 1.2 ¬Q, 1 ◊(Q ∨ S),
1.2 Q ∨ S   ⟨13⟩
_____
1.1 ¬P, 1.2 ¬Q, 1 ◊(Q ∨ S)   ⟨9⟩
_____
1 ◊(P ∨ R) ∧ ◊(Q ∨ S), 1.1 ¬P, 1.2 ¬Q   ⟨7⟩
_____
1 ◊(P ∨ R) ∧ ◊(Q ∨ S), 1 ¬◊Q, 1.1 ¬P   ⟨6⟩
_____
1 ◊(P ∨ R) ∧ ◊(Q ∨ S), 1 ¬◊P, 1 ¬◊Q   ⟨4, 5⟩
_____
1 ¬(◊P ∧ ◊Q), 1 ◊(P ∨ R) ∧ ◊(Q ∨ S)   ⟨2, 3⟩
_____
1 (◊P ∧ ◊Q) ⊃ (◊(P ∨ R) ∧ ◊(Q ∨ S))   ⟨1⟩

Figure 2: K Prefix Sequent Proof Example

line 10 produced lines 11 and 12 via an α rule. Prefix sequent $\langle 8 \rangle$ follows from $\langle 10 \rangle$ by a π rule application, with $\pi = \Diamond(P \vee R)$. Note that this application 'eliminates' $1.1\,P \vee R$, and the prefix 1.1 does occur in the consequent, as required. In the original tableau line 8 produced line 10 using the ν rule, and 1.1 was not new on the branch. Other steps have justifications that are similar. We leave this to the reader.

5 Translating To Nested Sequents

We now show how to translate prefix sequents into nested sequents. This translation, in its dual form, appeared in essence in [11, Chapter 8, Section 12] but without the syntactical machinery of nested sequents, which had not yet been introduced in the literature. The idea is very simple. Recall, all prefixes are required to begin with 1, and this plays a role in the translation. Formulas with the same prefix go in a single nested sequent. Formulas prefixed with 1 itself constitute the members of the top level nested sequent, and otherwise the nested sequent of formulas originally prefixed with $\sigma.n$ should appear as a member of the nested sequent of formulas originally prefixed with σ. Here is the formal version. In the course of it we allow the empty prefix, and identify a formula having an empty prefix with the formula itself. We also allow prefixes that begin with something other than 1.

Definition 5.1 Let S be a set of prefixed formulas, allowing the empty prefix and also prefixes starting with something other than 1.

1. For each positive integer n let $S^n = \{\sigma X \mid n.\sigma X \in S\}$, where $n.\sigma$ is the prefix σ with n added at the beginning.

2. Let $S^\circ = \{X \in S \mid X \text{ has the empty prefix}\} \cup \{(S^n)^\circ \mid S^n \neq \emptyset\}$.

Now, for a set S of prefixed formulas where no prefix is empty and all prefixes starti with 1, the *nested sequent translate* of S is $\mathcal{N}(S) =_{df} (S^1)^\circ$.

To illustrate this procedure we apply it to the set

$$S = \{1\,A, 1.1\,B, 1.1.1\,C, 1.1.2\,D, 1.2\,E, 1.2.1\,F\}.$$

$$\dfrac{\begin{array}{l}\Diamond(P\vee R),\,\big[\neg P,P,R\big],\\ \big[\neg Q\big]\quad\langle 11,12\rangle\end{array}}{\begin{array}{l}\Diamond(P\vee R),\,\big[\neg P,P\vee R\big],\\ \big[\neg Q\big]\quad\langle 10\rangle\end{array}}\qquad\dfrac{\begin{array}{l}\Diamond(Q\vee S),\,\big[\neg P\big],\\ \big[\neg Q,Q,S\big]\quad\langle 14,15\rangle\end{array}}{\begin{array}{l}\Diamond(Q\vee S),\,\big[\neg P\big],\\ \big[\neg Q,Q\vee S\big]\quad\langle 13\rangle\end{array}}$$

$$\dfrac{\Diamond(P\vee R),\,\big[\neg P\big],\big[\neg Q\big]\quad\langle 8\rangle\qquad\qquad\Diamond(Q\vee S),\,\big[\neg P\big],\big[\neg Q\big]\quad\langle 9\rangle}{\dfrac{\Diamond(P\vee R)\wedge\Diamond(Q\vee S),\,\big[\neg P\big],\big[\neg Q\big]\quad\langle 7\rangle}{\dfrac{\Diamond(P\vee R)\wedge\Diamond(Q\vee S),\neg\Diamond Q,\big[\neg P\big]\quad\langle 6\rangle}{\dfrac{\Diamond(P\vee R)\wedge\Diamond(Q\vee S),\neg\Diamond P,\neg\Diamond Q\quad\langle 4,5\rangle}{\dfrac{\neg(\Diamond P\wedge\Diamond Q),\Diamond(P\vee R)\wedge\Diamond(Q\vee S)\quad\langle 2,3\rangle}{(\Diamond P\wedge\Diamond Q)\supset(\Diamond(P\vee R)\wedge\Diamond(Q\vee S))\quad\langle 1\rangle}}}}}$$

Figure 3: K Nested Sequent Proof Example

where the letters stand for formulas.

$$S^1=\{A,1\,B,1.1\,C,1.2\,D,2\,E,2.1\,F\}$$
$$(S^1)^\circ=\{A,((S^1)^1)^\circ,((S^1)^2)^\circ\}=\{A,\{B,1\,C,2\,D\}^\circ,\{E,1\,F\}^\circ\}$$
$$\{B,1\,C,2\,D\}^\circ=\{B,\{C\}^\circ,\{D\}^\circ\}=\{B,\{C\},\{D\}\}$$
$$\{E,1\,F\}^\circ=\{E,\{F\}^\circ\}=\{E,\{F\}\}$$
$$\text{so}$$
$$\mathcal{N}(S)=(S^1)^\circ=\{A,\{B,\{C\},\{D\}\},\{E,\{F\}\}\}$$

Written using boxed sequent notation, $\mathcal{N}(S)$ is $A,[B,[C],[D]],[E,[F]]$.

Next we examine the effect of this translation on the Prefix Sequent Rules from Section 4. Consider the Prefix Sequent α Rule first, telling us we can conclude $S,\sigma\,\alpha$ from $S,\sigma\,\alpha_1$ and $S,\sigma\,\alpha_2$. Since formulas with the same prefixes wind up in the same subsequent of a nested sequent, this rule converts into one that tells us we may conclude $\Gamma(\alpha)$ if we have $\Gamma(\alpha_1)$ and $\Gamma(\alpha_2)$, which is the Nested Sequent α Rule from Section 3.

Next consider the Prefix Sequent ν Rule, telling us we may conclude $S,\sigma\,\nu$ from $S,\sigma.n\,\nu_0$ provided $\sigma.n$ does not occur in S. Since ν_0 is prefixed with $\sigma.n$ and this can't occur in S, under the translation ν_0 must be in a boxed sequent with nothing else in it. And since $\sigma.n$ extends σ, the boxed sequent containing ν_0 must be inside the one that will contain ν. Then the Prefix Sequent ν Rule converts to a rule that tells us if we have $\Gamma([\nu_0])$ we may derive $\Gamma(\nu)$, which is exactly the Nested Sequent ν Rule from Section 3.

Similarly each of the Prefix Sequent rules converts to the corresponding Nested Sequent rule.

Continuing with the example we have been following, the prefix sequent proof of Figure 2 converts into the proof in Figure 3, in the nested sequent system.

6   Going the Other Way

Nested sequent proofs translate back into prefixed tableau proofs as well, but in this direction the problems are greater. We first convert a nested sequent proof into a prefix sequent one, then we invert this to produce a prefixed tableau proof. The inversion process is straightforward. The problem lies with the conversion from nested sequents to prefix sequents. The translation from a prefix sequent into a nested sequent does not have a well-defined inverse—for example, both

$\{1\,X, 1.1\,Y, 1.2\,Z\}$ and $\{1\,X, 1.1\,Z, 1.2\,Y\}$ translate to the nested sequent $\{X, \{Y\}, \{Z\}\}$, or equivalently $X, [Y], [Z]$, so there is no unique backward translation. This means we cannot simply convert a nested sequent proof into a prefix sequent proof line by line, treating each step independently of the others. We must also make sure we have picked prefixed translates in such a way that they fit together to form a prefix tableau proof. We must translate entire proofs, not lines of proofs. In order to do this we introduce some additional machinery.

A subsequent of a nested sequent Γ may occur more than once in Γ. Since we are treating sequents as sets, something like $\{A, \{B\}, \{B\}\}$ is not an example of this, since it is simply equal to $\{A, \{B\}\}$, but the nested sequent $\{A, \{B, \{C\}\}, \{C\}\}$ does provide an example—in it $\{C\}$ has two occurrences, and these do not collapse into one. We want to *annotate* the *occurrences* in a nested sequent so that different occurrences of the same sequent have different annotations, and we will use prefixes as annotations. Formally, an annotation is a mapping from the set of subsequent occurrences in a nested sequent to prefixes, meeting certain conditions. Informally, we will indicate an annotation by writing subscripts on subsequent occurrences.

The conditions that must be met by an annotation of nested sequent Γ are these. All subsequent occurrences receive an annotation. Different subsequent occurrences have different annotations. Γ itself is annotated with 1. If Δ and Δ' are subsequent occurrences in Γ, Δ is annotated with σ, and $\Delta' \in \Delta$, then Δ' is annotated with $\sigma.n$ for some positive integer n.

As an example, consider again the nested sequent $\Gamma = \{A, \{B, \{C\}\}, \{C\}\}$. Here is an annotation of it: $\{A, \{B, \{C\}_{1.3.2}\}_{1.3}, \{C\}_{1.2}\}_1$. Obviously many annotations are possible.

If we have an annotated nested sequent Γ, it induces a prefixed sequent S in a straightforward way. Put $\sigma\,X$ in S if there is a subsequent occurrence Δ of Γ, annotated with σ, with X directly in Δ. For example, the annotated sequent $\{A, \{B, \{C\}_{1.3.2}\}_{1.3}, \{C\}_{1.2}\}_1$ induces the prefixed sequent $S = \{1\,A, 1.3\,B, 1.3.2\,C, 1.2\,C\}$. Notice that under the translation of Definition 5.1, the translate of S is the nested (unannotated) sequent we started with, $\{A, \{B, \{C\}\}, \{C\}\}$. This always happens, and we use the fact throughout.

Now we describe how to translate nested sequent proofs into prefix sequent proofs. We associate with each sequent in the nested sequent proof an annotation—the corresponding induced prefix sequents make up the prefix sequent proof. The process works from bottom up. Suppose we have a nested sequent proof, $\mathcal{N}$, of a nested sequent Ω. We construct a corresponding prefix sequent proof $\mathcal{P}$ as follows.

The last line of $\mathcal{N}$ is Ω. In general, many annotations of Ω are possible—pick one of them. Say the corresponding induced prefix sequent is S; we make S the last line of $\mathcal{P}$.

Next, suppose Γ occurs as a sequent in $\mathcal{N}$, and an annotation has been associated with Γ. There are two primary cases: Γ is an axiom, or Γ is not. Suppose first that Γ is an axiom, say $\Gamma(A, \neg A)$ where A is a propositional letter. Under the annotation, both A and $\neg A$ will occur directly in some annotated subsequent, say with annotation σ. Then the induced prefix sequent for Γ will have $\sigma\,A$ and $\sigma\,\neg A$ in it and hence will be a Prefix Sequent Axiom.

Finally, the serious case. Suppose Γ occurs as a sequent in $\mathcal{N}$, an annotation has been associated with Γ, and Γ is not an axiom. Then Γ follows from one, or two, nested sequents using one of the rules: double negation, α, β, ν, or π. We must associate annotations with the sequents from which Γ follows, and do so in a way that is compatible with the annotation associated with Γ. There is a case for each rule; we discuss the α rule and the ν rule as representative examples.

α **Rule** Γ is $\Gamma(\alpha)$ and follows in $\mathcal{N}$ from $\Gamma(\alpha_1)$ and $\Gamma(\alpha_2)$. $\Gamma(\alpha_1)$ is like $\Gamma(\alpha)$ except that an occurrence of α has been replaced with an occurrence of α_1. Carry this replacement out in the *annotated* version of $\Gamma(\alpha)$, and we get an appropriate annotation for $\Gamma(\alpha_1)$. Similarly for $\Gamma(\alpha_2)$. The corresponding induced prefix sequents will differ from the induced prefix sequent

associated with annotated $\Gamma(\alpha)$ in that, where the latter has $\sigma\,\alpha$, the former will have $\sigma\,\alpha_1$ and $\sigma\,\alpha_2$, for some prefix σ. There are no other differences. Clearly an application of Prefix Sequent α Rule has been generated. And the assignment of annotations has been carried one step upward.

ν **Rule** Γ is $\Gamma(\nu)$ and follows from $\Gamma([\nu_0])$ in $\mathcal{N}$. As in the previous case, $\Gamma([\nu_0])$ is like $\Gamma(\nu)$ except that an occurrence of ν has been replaced with $[\nu_0]$. And also, as in the previous case, if we carry out this replacement in the *annotated* version of $\Gamma(\nu)$ we *almost* have an annotated version of $\Gamma([\nu_0])$. What is missing is an annotation for the newly introduced subsequent $[\nu_0]$. But that is a member of another subsequent which already has an annotation, say σ. Annotate the new occurrence of $[\nu_0]$ with $\sigma.n$ where this is a prefix that does not otherwise occur in the annotated sequent. The annotated $\Gamma([\nu_0])$ induces a prefixed sequent from which the prefixed sequent induced by $\Gamma(\nu)$ follows by an application of the Prefix Sequent ν Rule, and once again we have extended the assignment of annotations one step upward.

Continuing in this way the entire nested sequent proof $\mathcal{N}$ is annotated, and induces a corresponding prefix sequent proof $\mathcal{P}$.

Here is a small example illustrating the ν Rule step above. Using the Nested Sequent ν Rule, the following is an allowed step in a proof, written in both boxed sequent and set notation.

$$\frac{A, [\Box B], [[B], C]}{A, [\Box B], [\Box B, C]} \qquad \frac{\{A, \{\Box B\}, \{\{B\}, C\}\}}{\{A, \{\Box B\}, \{\Box B, C\}\}}$$

Suppose we have as annotation for the conclusion, $\{A, \{\Box B\}_{1.2}, \{\Box B, C\}_{1.4}\}_1$, which induces the corresponding prefix sequent $\{1\,A, 1.2\,\Box B, 1.4\,\Box B, 1.4\,C\}$. The occurrence of $\Box B$ introduced by the sequent rule application is the one in the subsequent annotated with 1.4. We replace that occurrence with $\{B\}$ and annotate this new subsequent with $1.4.n$ where this must not otherwise occur; say we use $1.4.1$. This gives us the following annotated version of the premise sequent: $\{A, \{\Box B\}_{1.2}, \{\{B\}_{1.4.1}, C\}_{1.4}\}_1$. The prefix sequent induced by this is $\{1\,A, 1.2\,\Box B, 1.4.1\,B, 1.4\,C\}$, from which $\{1\,A, 1.2\,\Box B, 1.4\,\Box B, 1.4\,C\}$ follows using the Prefix Sequent ν Rule.

7   Other Modal Logics

We have been discussing the logic K. Other standard modal logics require additional rules. The original prefixed tableau rules from [10] covered many standard modal logics but they were not modular. The current commonly used rules, which are modular, come from [18] and are discussed further in [15, 19]. Rules for a number of logics were given, but we only discuss some of them here. Nested sequent rules for standard modal logics were formulated in [4]. For the most part the translations discussed above show the corresponding rules in the two systems are equivalent. The exceptions concern KD and S5, and we discuss these after other rules have been covered for both systems.

We begin with prefixed tableau rules. In the Prefixed Tableau 4 Rule it is required that $\sigma.n$ already occurs on the tableau branch. In the Prefixed Tableau D Rule if ν is $\Box X$ then π is $\Diamond X$, and if ν is $\neg\Diamond X$ then π is $\neg\Box X$. These rules are single-use, except for **4**, but this does not really matter for present purposes. Since the Prefixed Tableau ν Rule also applies to the premise formulas below, they cannot be removed from consideration when one of the rules below has been applied.

Prefixed Tableau T Rule	$\dfrac{\sigma\,\nu}{\sigma\,\nu_0}$
Prefixed Tableau D Rule	$\dfrac{\sigma\,\nu}{\sigma\,\pi}$
Prefixed Tableau B Rule	$\dfrac{\sigma.n\,\nu}{\sigma\,\nu_0}$
Prefixed Tableau 4 Rule	$\dfrac{\sigma\,\nu}{\sigma.n\,\nu}$
Prefixed Tableau 4r Rule	$\dfrac{\sigma.n\,\nu}{\sigma\,\nu}$

The following chart specifies which rules must be added to those for K to get tableau systems for various common logics.

Logic	Special Rules
KD	**Prefixed Tableau Rule D**
KD4	**Prefixed Tableau Rules D, 4**
T	**Prefixed Tableau Rule T**
K4	**Prefixed Tableau Rule 4**
KB	**Prefixed Tableau Rule B**
KDB	**Prefixed Tableau Rules D, B**
B	**Prefixed Tableau Rules B, T**
S4	**Prefixed Tableau Rules T, 4**
S5	**Prefixed Tableau Rules T, 4, 4r**

There are some peculiarities. It was pointed out in [18] that though one might expect that Prefixed Tableau Rules **T**, **4**, and **B** would combine to give a system for S5, they do not. A similar thing happens with nested sequents, and was noted in [4, 5].

Next we give the nested sequent rules from [4], except for a rule called **5** which we discuss later. Once again we do not assume formulas are in negation normal form, and we make use of uniform notation. We recall that $[\ldots]$ is a *non-empty* nested sequent, and $[Z, \ldots]$ is the same sequent but with Z added.

Nested Sequent t Rule	$\dfrac{\Gamma(\pi, \pi_0)}{\Gamma(\pi)}$
Nested Sequent d Rule	$\dfrac{\Gamma(\pi, [\pi_0])}{\Gamma(\pi)}$
Nested Sequent b Rule	$\dfrac{\Gamma(\pi_0, [\pi, \ldots])}{\Gamma([\pi, \ldots])}$
Nested Sequent 4 Rule	$\dfrac{\Gamma(\pi, [\pi, \ldots])}{\Gamma(\pi, [\ldots])}$

As we saw earlier, tableau rules translate into nested sequent rules, and conversely. Let us carry the process out with the Prefixed Tableau 4 Rule, as an example. First of all, we make it a bit less schematic by 'filling in' the implied rest of the branch, as we did in the discussion of Section 4. The rule becomes: a branch containing $S, \sigma\,\nu$ may be extended to one containing $S, \sigma\,\nu, \sigma.n\,\nu$, provided the prefix $\sigma.n$ was already on the branch. Next this rule, which is part of a refutation system, is inverted to become a forward reasoning rule. In doing so formulas have their polarities switched, and so ν formulas become π formulas and conversely. The original rule becomes the following,

where the condition that $\sigma.n$ already occur becomes the requirement that it must occur in the consequent, below the line, which in effect means that $\sigma.n$ must occur in S.

Prefix Sequent 4 Rule
$$\frac{S, \sigma\,\pi, \sigma.n\,\pi}{S, \sigma\,\pi}$$

Finally we translate this into nested sequent form. Once again, prefix extensions correspond to nesting of sequents. The condition that $\sigma.n$ occur in S turns into the condition that the sequent corresponding to $\sigma.n$ has something else in it besides π, so that the sequent still appears in the consequent once the counterpart of $\sigma.n\,\pi$ has been removed. The translated rule is the following, which in fact is the Nested Sequent 4 Rule stated above.

Nested Sequent 4 Rule
$$\frac{\Gamma(\pi, [\pi, \ldots])}{\Gamma(\pi, [\ldots])}$$

The correspondence continues to the **t** and **b** rules as well. The Prefixed Tableau D Rule works correctly, but is a little unsatisfying in that it violates the subformula principle that one expects from tableaus. Of course it does so in a minor way, but even this can be avoided. If we translate the Nested Sequent d Rule back into a tableau rule we get the rule immediately below. It looks just like the Prefixed Tableau ν Rule from Section 2, but now the side condition is dropped. We no longer require that $\sigma.n$ already be on the branch; it may also be new. This rule has, in fact, been used in the tableau community.

Prefixed Tableau D Rule
$$\frac{\sigma\,\nu}{\sigma.n\,\nu_0}$$

Capturing the logic S5 is more complicated, apparently. In [4] this makes use of a rule called **5**. The formulation of the nested sequent **5** rule involves two 'holes', something we have not considered here. It is also shown to be derivable from three 'single hole' rules called, **5a**, **5b**, and **5c**. On the other hand, the tableau rule **4r** is simply the **4** rule upside down and it, in conjunction with **T** and **4** rules captures S5. It is of interest, then, to see what this rule becomes in a nested sequent system. The Prefixed Tableau 4 Rule corresponds to the Nested Sequent 4 Rule and this rule, roughly speaking, allows us to eliminate a π occurrence that is inside a boxed sequent. In a similar way the Prefixed Tableau 4r Rule, which corresponds to the Euclidean condition, translates to the following, which allows us to eliminate a π occurrence outside a boxed sequent.

Nested Sequent 4r Rule
$$\frac{\Gamma(\pi, [\pi, \ldots])}{\Gamma([\pi, \ldots])}$$

Adding **Nested Sequent Rules t**, **4**, and **4r** to the basic system for K produces a system for S5. The exact relationship between **4r** and **5** might be interesting to explore.

8 Quantification

Discussions of nested sequent systems for quantified modal logics are relatively rare. On the other hand, quantification is well-represented using prefixed tableau systems—[14] is a good example. Since the translation procedures work just as well when quantifiers are involved, we can make use of them to generate nested sequent systems for modal logics that admit quantification, and we sketch how in this section.

As it happens, quantification in modal logic is a rich subject. One can quantify over objects or over intensions. Both versions have been explored via prefixed tableau systems, see [12] for instance. We do not consider intensional quantifiers here. But even for object quantification, quantifiers

can be *actualist* or *possibilist*, or something in between. In terms of Kripke semantics, actualist quantification is *varying domain* while possibilist quantification is *constant domain*. One can also impose specialized relationships on varying domain semantics, *monotonicity* or *anti-monotonicity*. All these can be captured using prefixed-based systems.

In this section we first sketch a prefixed tableau system for K with possibilist quantifiers—constant domain. This is the simplest to treat. We then present the corresponding nested sequent system. Direct prefixed tableau systems for varying domain/actualist semantics are more complex as presented in [14], and adding monotonicity or anti-monotonicity conditions complicates things even more. Instead we take an alternate route, also outlined in [14], involving the introduction of an *existence* predicate. We discuss this more fully below.

First, of course, the language must be elaborated. From now on we assume we have *relation symbols* of various arities, variables, and quantifiers, $\forall$ and $\exists$. To keep things as simple as possible, we do not have constant or function symbols. Formulas are built up in the standard way and we omit the details.

When working with first-order formulas one must be careful about instantiating a quantifier with a variable that may be captured by other quantifiers. In [24] a simple device is used to avoid the problem, and we adopt it here. In addition to the basic list of individual variables, a second list is introduced, called *parameters*, solely for use in proofs—that is, they do not appear in formulas being proved. In particular, they never appear quantified. Then instantiating a quantifier with a parameter never gives rise to problems of inadvertent capture.

We also adopt uniform notation for quantifiers, from [24], with γ for universals and δ for existentials. The following table gives the cases, along with *instances*. In it, a is an arbitrary parameter.

γ	$\gamma(a)$		δ	$\delta(a)$
$(\forall x)\varphi(x)$	$\varphi(a)$		$(\exists x)\varphi(x)$	$\varphi(a)$
$\neg(\exists x)\varphi(x)$	$\neg\varphi(a)$		$\neg(\forall x)\varphi(x)$	$\neg\varphi(a)$

In possibilist quantification models there is a single domain that quantifiers range over, independent of possible worlds. One can think of its members as the *possibly existing objects*. A discussion of motivation can be found in [14, 16]. The tableau rules for possibilist quantification are quite simple, and are as follows.

γ **Rule** A branch containing $\sigma\,\gamma$ may be extended with a node labeled $\sigma\,\gamma(a)$ for any parameter a.

δ **Rule** A branch containing $\sigma\,\delta$ may be extended with a node labeled $\sigma\,\delta(a)$, where a is a parameter that is new to the branch.

Schematically, these rules are as follows. The δ rule is single-use, and if the δ rule applies to a prefixed formula no other rule does, so a δ prefixed formula can be removed after the rule is applied to it. The γ rule is not single-use.

Prefixed Tableau Possibilist γ Rule

$$\frac{\sigma\,\gamma}{\sigma\,\gamma(a)}$$
any parameter a

Prefixed Tableau Possibilist δ Rule

$$\frac{\sigma\,\delta}{\sigma\,\delta(a)}$$
new parameter a

$$1 \quad \neg[(\forall x)\Box P(x) \supset \Box(\forall x)P(x)] \quad \langle 1 \rangle$$
$$1 \quad (\forall x)\Box P(x) \quad \langle 2 \rangle$$
$$1 \quad \neg\Box(\forall x)P(x) \quad \langle 3 \rangle$$
$$1.1 \, \neg(\forall x)P(x) \quad \langle 4 \rangle$$
$$1.1 \, \neg P(a) \quad \langle 5 \rangle$$
$$1 \quad \Box P(a) \quad \langle 6 \rangle$$
$$1.1 \quad P(a) \quad \langle 7 \rangle$$

Figure 4: Possibilist K Prefixed Tableau Proof Example

$$\frac{\neg(\forall x)\Box P(x), \neg\Box P(a), [P(a), \neg P(a)] \quad \langle 7 \rangle}{\frac{\neg(\forall x)\Box P(x), \neg\Box P(a), [P(a)] \quad \langle 6 \rangle}{\frac{\neg(\forall x)\Box P(x), [P(a)] \quad \langle 5 \rangle}{\frac{\neg(\forall x)\Box P(x), [(\forall x)P(x)] \quad \langle 4 \rangle}{\frac{\neg(\forall x)\Box P(x), \Box(\forall x)P(x) \quad \langle 2, 3 \rangle}{(\forall x)\Box P(x) \supset \Box(\forall x)P(x) \quad \langle 1 \rangle}}}}}$$

Figure 5: Possibilist K Nested Sequent Proof Example

Using these rules, Figure 4 shows a closed K tableau proof of the Barcan formula, $(\forall x)\Box P(x) \supset \Box(\forall x)P(x)$. In it, $\langle 2 \rangle$ and $\langle 3 \rangle$ are from $\langle 1 \rangle$ by α; $\langle 4 \rangle$ is from $\langle 3 \rangle$ by π; $\langle 5 \rangle$ is from $\langle 4 \rangle$ by δ; $\langle 6 \rangle$ is from $\langle 2 \rangle$ by γ; and $\langle 7 \rangle$ is from $\langle 6 \rangle$ by ν.

The tableau rules easily convert to nested sequent rules. We omit details. The results are as follows.

Nested Sequent Possibilist γ Rule $\qquad \dfrac{\Gamma(\gamma(a))}{\Gamma(\gamma)}$

Nested Sequent Possibilist δ Rule $\qquad \dfrac{\Gamma(\delta, \delta(a))}{\Gamma(\delta)}$

In the nested sequent γ rule, the parameter a must not occur in the consequent, that is, below the line. In the δ rule there are no restrictions.

The tableau proof of Figure 4 for the Barcan formula turns over and inverts to become a prefix sequent proof, which we omit. That, in turn, translates into the nested sequent proof shown in Figure 5, using the rules just stated along with the earlier ones. We leave the verification to you—steps have been numbered to match corresponding steps in the tableau of Figure 4.

In [14] the prefix tableau quantification rules are modified so that parameters too have prefixes—this makes possible a proof system for actualist/varying domain semantics. Then special rules concerning these prefixed parameters are introduced to handle monotonicity and anti-monotonicity conditions. All this works well, but there is an even more transparent way, sketched in [14] and given in a bit more detail here. We introduce an *existence predicate*, E, new to the language. Semantically, at each possible world of a constant domain Kripke model, interpret E to be true of those objects that 'actually' exist at that world. Relativize quantifiers to E, and then just use the possibilist rules above. The one additional rule we need embodies an assumption that at each possible world something actually exists, so local domains of quantification are non-empty. We begin with the familiar business of relativization.

Definition 8.1 We define a formula X^E for each formula X (not containing E) as follows.

1. If A is atomic, $A^E = A$;

2. $(\neg X)^E = \neg(X^E)$;

3. $(X \circ Y)^E = X^E \circ Y^E$, for each binary connective $\circ$;

4. $(mX)^E = m(X^E)$ for a modal operator m;

5. $[(\forall x)\varphi]^E = (\forall x)[E(x) \supset \varphi^E]$;

6. $[(\exists x)\varphi]^E = (\exists x)[E(x) \wedge \varphi^E]$.

As noted above, we need to ensure that actualist quantification domains are non-empty. In the following rule, the parameter a is new to the branch, while σ is a prefix that already occurs on the branch.

Prefix Tableau Existence Rule
$$\frac{}{\sigma\, E(a)}$$
$$\text{new parameter } a$$

Now, an actualist tableau proof of X is simply a possibilist tableau proof of X^E, allowing the Prefix Tableau Existence Rule. We give a proof example after monotonicity and anti-monotonicity have been discussed.

The Prefix Tableau Existence Rule converts to the following sequent rule. In it, the parameter a is not allowed to occur in the consequent, and Z represents either a formula or a nested sequent, the point being that $\neg E(a)$ must occur in a non-empty context since σ must already occur on the branch.

Nested Sequent Existence Rule
$$\frac{\Gamma(\neg E(a), Z)}{\Gamma(Z)}$$

Kripke varying domain models meet a *monotonicity* condition if, whenever possible world w_2 is accessible from w_1, any object actually existing at w_1 also actually exists at w_2. The *anti-monotonicity* condition is met if, whenever possible world w_2 is accessible from w_1, any object actually existing at w_2 also actually exists at w_1. Anti-monotonicity corresponds to Barcan formula validity, monotonicity corresponds to converse Barcan formula validity.

Capturing monotonicity and anti-monotonicity using an existence predicate is quite easy. Here are the tableau rules. In the monotonicity rule, $\sigma.n$ must already occur on the branch. The anti-monotonicity rule is single-use, but it is not the only rule that might be applied to $\sigma.n\, E(a)$; monotonicity might also apply, if present in the system. The monotonicity rule is not single-use.

Prefixed Tableau Monotonicity Rule
$$\frac{\sigma\, E(a)}{\sigma.n\, E(a)}$$
$$\sigma.n \text{ not new}$$

Prefixed Tableau Anti-Monotonicity Rule
$$\frac{\sigma.n\, E(a)}{\sigma\, E(a)}$$

As an example, we give a K tableau proof of an atomic version of the Barcan formula using the actualist quantification system, but assuming Anti-Monotonicity. The formula reads $(\forall x)\Box P(x) \supset \Box(\forall x)P(x)$, where P is a one-place relation symbol. Relativized to the existence predicate, it is

$$
\begin{array}{lll}
1 & \neg\{(\forall x)(E(x) \supset \Box P(x)) \supset \Box(\forall x)(E(x) \supset P(x))\} & \langle 1 \rangle \\
1 & (\forall x)(E(x) \supset \Box P(x)) & \langle 2 \rangle \\
1 & \neg\Box(\forall x)(E(x) \supset P(x)) & \langle 3 \rangle \\
1.1 & \neg(\forall x)(E(x) \supset P(x)) & \langle 4 \rangle \\
1.1 & \neg(E(a) \supset P(a)) & \langle 5 \rangle \\
1.1 & E(a) & \langle 6 \rangle \\
1.1 & \neg P(a) & \langle 7 \rangle \\
1 & E(a) \supset \Box P(a) & \langle 8 \rangle
\end{array}
$$

$$
\begin{array}{ll}
1\ \neg E(a) \quad \langle 9 \rangle & 1\ \ \Box P(a) \quad \langle 10 \rangle \\
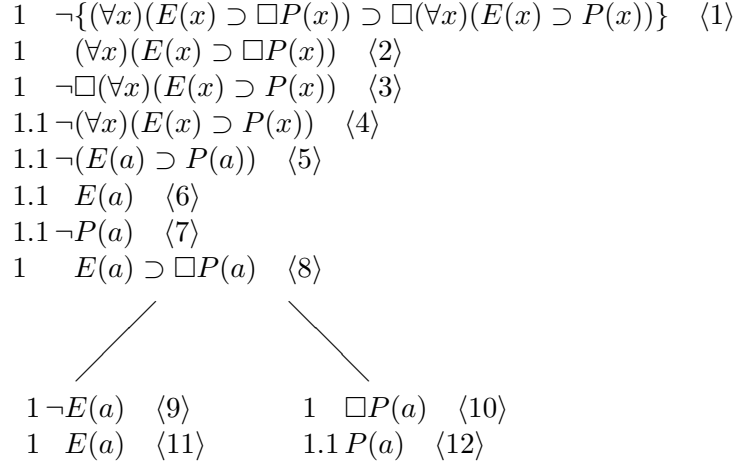1\ \ E(a) \quad \langle 11 \rangle & 1.1\ P(a) \quad \langle 12 \rangle
\end{array}
$$

Figure 6: Anti-Monotonic K Prefixed Tableau Example

$(\forall x)(E(x) \supset \Box P(x)) \supset \Box(\forall x)(E(x) \supset P(x))$, and we show a proof of this in Figure 6. In the proof, $\langle 2 \rangle$ and $\langle 3 \rangle$ are from $\langle 1 \rangle$ by α, $\langle 4 \rangle$ is from $\langle 3 \rangle$ by π, $\langle 5 \rangle$ is from $\langle 4 \rangle$ by δ, $\langle 6 \rangle$ and $\langle 7 \rangle$ are from $\langle 5 \rangle$ by α, $\langle 8 \rangle$ is from $\langle 2 \rangle$ by γ, $\langle 9 \rangle$ and $\langle 10 \rangle$ are from $\langle 8 \rangle$ by β, $\langle 11 \rangle$ is from $\langle 6 \rangle$ by anti-monotonicity, and $\langle 12 \rangle$ is from $\langle 10 \rangle$ by ν.

The tableau rules convert to the following nested sequent rules. We describe the conversion process for the Prefixed Tableau Anti-Monotonicity Rule, as we did earlier with the Prefixed Tableau 4 Rule, and in the future we skip such explanations. It converts a branch containing $S, \sigma.n\, E(a)$ to one containing $S, \sigma.n\, E(a), \sigma\, E(a)$. This inverts to yield the forward reasoning rule: from $S, \sigma.n\, \neg E(a), \sigma\, \neg E(a)$ conclude $S, \sigma.n\, \neg E(a)$. As stated formally, there are two versions of the nested sequent rule for anti-monotonicity, depending on whether or not the nested sequent containing $\neg E(a)$ has anything else in it or not. This duplication is an artifact of our convention that $[\ldots]$ stands for a non-empty nested sequent.

Nested Sequent Monotonicity Rule $\qquad \dfrac{\Gamma(\neg E(a), [\neg E(a), \ldots])}{\Gamma(\neg E(a), [\ldots])}$

Nested Sequent Anti-Monotonicity Rule $\quad \dfrac{\Gamma(\neg E(a), [\neg E(a), \ldots])}{\Gamma([\neg E(a), \ldots])}$

$$
\frac{\Gamma(\neg E(a), [\neg E(a)])}{\Gamma([\neg E(a)])}
$$

Figure 7 gives a nested sequent translate of the tableau proof shown in Figure 6, establishing the Barcan formula, given anti-monotonicity. As usual, lines are numbered to match those of the tableau proof.

A final observation. By relativizing some quantifiers to E but not others, one can mix actualist and possibilist quantification in the same formula, and proof rules need no modification.

$$\cfrac{\cfrac{\neg(\forall x)(E(x) \supset \Box P(x)), E(a),}{\neg E(a), [\neg E(a), P(a)] \quad \langle 11 \rangle}}{\cfrac{\neg(\forall x)(E(x) \supset \Box P(x)), E(a),}{[\neg E(a), P(a)] \quad \langle 9 \rangle}} \qquad \cfrac{\cfrac{\neg(\forall x)(E(x) \supset \Box P(x)), \neg\Box P(a),}{[\neg P(a), \neg E(a), P(a)] \quad \langle 12 \rangle}}{\cfrac{\neg(\forall x)(E(x) \supset \Box P(x)), \neg\Box P(a),}{[\neg E(a), P(a)] \quad \langle 10 \rangle}}$$

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{\neg(\forall x)(E(x) \supset \Box P(x)), \neg(E(a) \supset \Box P(a)), [\neg E(a), P(a)] \quad \langle 8 \rangle}}{\neg(\forall x)(E(x) \supset \Box P(x)), [\neg E(a), P(a)] \quad \langle 6, 7 \rangle}}{\neg(\forall x)(E(x) \supset \Box P(x)), [E(a) \supset P(a)] \quad \langle 5 \rangle}}{\neg(\forall x)(E(x) \supset \Box P(x)), [(\forall x)(E(x) \supset P(x))] \quad \langle 4 \rangle}}{\cfrac{\neg(\forall x)(E(x) \supset \Box P(x)), \Box(\forall x)(E(x) \supset P(x)) \quad \langle 2, 3 \rangle}{(\forall x)(E(x) \supset \Box P(x)) \supset \Box(\forall x)(E(x) \supset P(x)) \quad \langle 1 \rangle}}$$

Figure 7: Anti-Monotonic K Nested Sequent Example

9 Justification Logics and **S4LP**

A relatively recent development in modal logics is *justification logics*. In these, $\Box X$ is replaced with $t{:}X$, where t is one of an infinite family of justification terms. Informally, while $\Box X$ simply asserts that X is necessary, $t{:}X$ asserts that X is necessary for reason t. The article [2] is probably the most accessible survey of the subject. The first of the justification logics was LP, in which justification terms were intended to abstractly represent mathematical proofs, [1]. Justification logics correspond to standard modal logics via what are called *Realization Theorems*, which we do not go into here, but in this sense LP corresponds to the modal logic S4. There is a semantics for LP, [9], which adds machinery to the standard Kripke possible world semantics. In [3], a logic incorporating both standard modal operators and justification terms was introduced, called S4LP. Since LP models include all the machinery of S4 models, they also serve to model the combined logic S4LP. An axiom system was given for S4LP in [3] and proved sound; completeness was shown in [8]. Tableau/Gentzen systems were more of a problem. In [8] a system was given that still needed one axiom, an unsatisfactory feature. This was eliminated in [22], but that turned out to implicitly assume cut in its completeness proof. This is being corrected in [23], which is not yet published.

S4LP is one of several logics that combine *implicit* modal operators, the standard $\Box$ and $\Diamond$, with *explicit* justification terms. We conclude this paper with a discussion of S4LP, and some related implicit/explicit logics. We discuss semantics only briefly, and do not discuss axiomatics at all. The new material in this paper is the nested sequent systems for these logics. In turn they derive from prefixed tableau systems that were developed in [17].

10 Justification Logic Syntax and Semantics

As noted, the justification logics we consider here contain both *implicit* modal operators, the usual $\Box$ and $\Diamond$, and *explicit* operators, justification terms. Formulas are built up from propositional letters using $\land$, $\lor$, $\neg$, $\supset$, $\Box$ and $\Diamond$ in the usual way. Justification Logic adds to this a set of *justification terms*. These are formed from *constants* and *variables* using certain operations. These operations always include $+$ and $\cdot$, so that if s and t are justification terms, so are $(s + t)$ and $(s \cdot t)$. Our logics here will sometimes include $!$; if this is present, then if s is a justification term so is $!s$. The final rule of formation for our logics is: if X is a formula and t is a justification term, then $t{:}X$ is a formula.

The formula $t{:}X$ should be thought of as asserting that X is true for reason t. Justification

constants are intended to represent logical facts that are not analyzed further—typically axioms. Justification variables stand for unspecified justifications. $(s + t)$ is supposed to justify whatever is justified by s together with whatever is justified by t. If t justifies X and s justifies $X \supset Y$ then $(s \cdot t)$ is supposed to justify Y, and so $\cdot$ corresponds to *modus ponens*. The operator ! is a positive justification checker; if t justifies X then $!t$ justifies the fact that t justifies X.

Note that the sublanguage with no justification terms is just the conventional modal language. Likewise the sublanguage with no modal operators is the language usually used for justification logics. The overall language simply combines these two—we sometimes refer to it as an *implicit/explicit language*, and similarly for logics that use the language. In axiomatic developments of justification logics one commonly takes $\perp$ as primitive and $\supset$ as the only propositional connective. In the present approach there is no particular advantage to this.

There are two families of models for the implicit/explicit logics considered here: Fitting models and Artemov-Fitting models. Either will serve for present purposes—we use Fitting models, which involve less machinery, though the difference is not major unless a multi-agent version is considered, which is not the case here.

A (Fitting) *model* is a structure $\mathcal{M} = \langle \mathcal{G}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$ meeting the following conditions. $\langle \mathcal{G}, \mathcal{R} \rangle$ is a standard frame, where $\mathcal{G}$ is a set of possible worlds, technically just a non-empty set, and $\mathcal{R}$ is a binary relation on it. $\mathcal{V}$ is a mapping from propositional variables to subsets of $\mathcal{G}$, specifying atomic truth at possible worlds. $\mathcal{E}$ is an *evidence function*, which originated in [20]. It maps justification terms and formulas to sets of worlds. The idea is, if the possible world Γ is in $\mathcal{E}(t, X)$ then t is *relevant* or *admissible* evidence for X at world Γ.

Given a model $\mathcal{M} = \langle \mathcal{G}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$, truth of formula X at possible world Γ is denoted by $\mathcal{M}, \Gamma \Vdash X$, and is required to meet the following conditions, which are a mixture of familiar modal conditions and those for pure justification logics. For each $\Gamma \in \mathcal{G}$:

1. $\mathcal{M}, \Gamma \Vdash P \Longleftrightarrow \Gamma \in \mathcal{V}(P)$ for P a propositional letter;

2. $\mathcal{M}, \Gamma \Vdash \neg X \Longleftrightarrow \mathcal{M}, \Gamma \nVdash X$;

3. $\mathcal{M}, \Gamma \Vdash X \wedge Y \Longleftrightarrow \mathcal{M}, \Gamma \Vdash X$ and $\mathcal{M}, \Gamma \Vdash Y$;

4. $\mathcal{M}, \Gamma \Vdash X \vee Y \Longleftrightarrow \mathcal{M}, \Gamma \Vdash X$ or $\mathcal{M}, \Gamma \Vdash Y$;

5. $\mathcal{M}, \Gamma \Vdash X \supset Y \Longleftrightarrow \mathcal{M}, \Gamma \nVdash X$ or $\mathcal{M}, \Gamma \Vdash Y$;

6. $\mathcal{M}, \Gamma \Vdash \Box X \Longleftrightarrow \mathcal{M}, \Delta \Vdash X$ for every $\Delta \in \mathcal{G}$ such that $\Gamma \mathcal{R} \Delta$;

7. $\mathcal{M}, \Gamma \Vdash \Diamond X \Longleftrightarrow \mathcal{M}, \Delta \Vdash X$ for some $\Delta \in \mathcal{G}$ such that $\Gamma \mathcal{R} \Delta$;

8. $\mathcal{M}, \Gamma \Vdash (t{:}X) \Longleftrightarrow \Gamma \in \mathcal{E}(t, X)$ and, for every $\Delta \in \mathcal{G}$ with $\Gamma \mathcal{R} \Delta$, $\mathcal{M}, \Delta \Vdash X$.

The first seven items above are standard to modal semantics. The final justification condition could also have been stated as

$$\mathcal{M}, \Gamma \Vdash (t{:}X) \Longleftrightarrow \Gamma \in \mathcal{E}(t, X) \text{ and } \mathcal{M}, \Gamma \Vdash \Box X.$$

The idea is that $\mathcal{M}, \Gamma \Vdash t{:}X$ if $\Box X$ is true at Γ and at Γ it is the case that t is a relevant justification for X.

Some conditions must be placed on evidence functions. The ones common to all justification logics are the following:

9. $\mathcal{E}(s, X \supset Y) \cap \mathcal{E}(t, X) \subseteq \mathcal{E}(s \cdot t, Y)$;

10. $\mathcal{E}(s, X) \cup \mathcal{E}(t, X) \subseteq \mathcal{E}(s + t, X)$.

Thus far we have defined what are called J_0 models. As usual, a formula is *valid* (in the class of J_0 models, or in some specified subclass) if it is true at every possible world of every model. Item 9 above ensures the validity of $s{:}(X \supset Y) \supset (t{:}X \supset [s \cdot t]{:}Y)$, making $\cdot$ correspond to *modus ponens*, as noted earlier. Likewise item 10 ensures the validity of both $s{:}X \supset [s + t]{:}X$ and $t{:}X \supset [s + t]{:}X$.

Constant Specifications can now be brought into the picture. Constants are used to represent justifications of assumptions when these are not analyzed any further. Suppose we postulate that some validity A is justified, but we do not intend to analyze the justification. We simply postulate $e_1 : A$ for some justification constant e_1. If, furthermore, we want to postulate that this new principle $e_1{:}A$ is also justified, we can postulate $e_2{:}(e_1{:}A)$ for a constant e_2. And so on. The set of assumptions of this kind is called a *Constant Specification*. More formally, a *Constant Specification* CS is a set of formulas of the form $e_n : e_{n-1} : \ldots : e_1 : A \quad (n \geq 1)$, where A is valid formula, and $e_1, e_2, \ldots, e_n$ are constants. It is assumed that CS contains all intermediate specifications, i.e., whenever $e_n : e_{n-1} : \ldots : e_1 : A$ is in CS, then $e_{n-1} : \ldots : e_1 : A$ is in CS too. Typically constants are associated with axioms of some axiomatic formulation. Details are not relevant to our work here, and are omitted.

A J_{CS} model is a J_0 model in which all members of constant specification CS are valid. We say such a model *meets* the constant specification. The empty set is a constant specification, and $\mathsf{J}_0 = \mathsf{J}_\emptyset$.

J_{CS} models were created for justification logics without implicit modalities, but since these models have all the usual machinery of Kripke models, they provide an interpretation for the usual modal operators $\Box$ and $\Diamond$ as well as for justification terms. We made this explicit in our specification of truth at a world, above. Then the J_{CS} models of this section provide a semantics for an implicit/explicit combination of the modal logic K and the justification logic known as J; we refer to the combination as $\mathsf{K} + \mathsf{J}$.

11   Prefixed Tableaus for $\mathsf{K} + \mathsf{J}$

We gave prefixed tableau rules for K in Section 2, and these can be extended to $\mathsf{K} + \mathsf{J}$. The system here is a subsystem of that for $\mathsf{S4LPN}$ in [17]. To the earlier rules for K are added the following rules for justification terms.

Prefix Tableau Application Rule
$$\frac{\sigma \, \neg(t \cdot s){:}Y}{\sigma \, \neg t{:}(X \supset Y) \;\big|\; \sigma \, \neg s{:}X}$$

Prefix Tableau Weakening Rules
$$\frac{\sigma \, \neg(s + t){:}X}{\sigma \, \neg s{:}X}$$

$$\frac{\sigma \, \neg(s + t){:}X}{\sigma \, \neg t{:}X}$$

Prefix Tableau Connecting Rule
$$\frac{\sigma \, t{:}X}{\sigma.n \, X}$$
$\sigma.n$ not new

The Prefix Tableau Application Rule is not single-use because it can be applied to the same prefixed formula, involving Y, with different choices for X. Likewise the Prefix Tableau Connecting Rule is not single-use. The Prefix Tableau Weakening Rule is single-use, but more than one rule applies to the premise of each of the rules, namely the other one.

$$
\begin{array}{lll}
1 & \neg[x{:}F \supset \Box F] & \langle 1 \rangle \\
1 & x{:}F & \langle 2 \rangle \\
1 & \neg\Box F & \langle 3 \rangle \\
1.1 & \neg F & \langle 4 \rangle \\
1.1 & F & \langle 5 \rangle
\end{array}
$$

Figure 8: K + J Tableau Example

$$
\begin{array}{ll}
1\,\neg[y{:}x{:}F \supset (c \cdot y){:}\Box F] & \langle 1 \rangle \\
1\;\;y{:}x{:}F & \langle 2 \rangle \\
1\,\neg(c \cdot y){:}\Box F & \langle 3 \rangle
\end{array}
$$

$$
1\,\neg c{:}[x{:}F \supset \Box F] \quad \langle 4 \rangle \qquad 1\,\neg y{:}x{:}F \quad \langle 5 \rangle
$$

Figure 9: A Second K + J Tableau Example

Even though the rules do not include Cut, the Prefix Tableau Application Rule tells us tableau proofs do not obey the subformula principle—this is not an analytic proof system.

The rules for termination of a proof need some modifications from K. Of course a tableau is *closed* if each branch is closed. But now a branch is closed for one of two reasons, each involving the presence of a syntactic contradiction. Let us say a formula is *generalized-atomic* if it is a propositional letter, or if it is of the form $t{:}X$. Then, a branch is closed if it contains $\sigma\, X$ and $\sigma\, \neg X$ for some prefix σ and some generalized-atomic formula X. But also, suppose we are giving a proof *assuming constant specification CS*; then a branch is closed if it contains $\sigma\, \neg t{:}X$, where $t{:}X \in CS$.

As usual, a closed tableau beginning with $1\,\neg X$ constitutes a proof of X, but we can also consider tableaus beginning with a more general set of prefixed formulas.

We give two simple tableau examples. The first is a proof of $x{:}F \supset \Box F$, in Figure 8. In this, 2 and 3 are from 1 by the α rule; 4 is from 3 by the π rule; 5 is from 2 by Connecting rule. Closure is by 4 and 5.

Once soundness of the tableau system has been proved, in Section 12, the proof in Figure 8 verifies that $x{:}F \supset \Box F$ is valid. Or validity can be confirmed directly. Now suppose we have a constant specification CS that provides a constant justification term for $x{:}F \supset \Box F$, specifically we assume $c{:}[x{:}F \supset \Box F] \in CS$. Assuming this constant specification, Figure 9 shows a tableau proof of $y{:}x{:}F \supset (c \cdot y){:}\Box F$. In Figure 9, $\langle 2 \rangle$ and $\langle 3 \rangle$ are from $\langle 1 \rangle$ by the α rule; $\langle 4 \rangle$ and $\langle 5 \rangle$ are from $\langle 3 \rangle$ by the application rule. Closure is by $\langle 2 \rangle$ and $\langle 5 \rangle$ on the right branch, and by $\langle 4 \rangle$ on the left branch, using the constant specification.

12 Soundness and Completeness

Soundness and completeness are proved for the stronger system S4LPN in [17], with respect to Artemov-Fitting models. We scale those proofs down to K + J to illustrate how these may be proved for prefixed tableau systems in general. We use the Fitting models, as described in Section 10, and not the more elaborate Artemov-Fitting models of [17]. One may also consult [13, Chapter 2, Section 6.2].

We begin with soundness which, as usual, is simpler. Call a set S of prefixed formulas *satisfiable* if there is some model $\mathcal{M} = \langle \mathcal{G}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$, and some mapping f from the prefixes appearing in S to the set $\mathcal{G}$, all meeting the following conditions.

- If $\sigma.n$ occurs in S (and hence also σ occurs), then $f(\sigma)\mathcal{R}f(\sigma.n)$.

- If σX is in S then $\mathcal{M}, f(\sigma) \Vdash X$.

Call a tableau *branch* satisfiable if the set of prefixed formulas on it is satisfiable. Call a *tableau* satisfiable if some branch is. Proofs of the following two Lemmas are straightforward, and are omitted.

Lemma 12.1 *If a tableau construction rule is applied to a satisfiable tableau, the result is another satisfiable tableau.*

Lemma 12.2 *A closed tableau is not satisfiable.*

It follows easily from these Lemmas that if there is a closed tableau for a set S, then S is not satisfiable. In particular, we have the following.

Theorem 12.3 *If X has a tableau proof assuming constant specification* CS*, then X is valid in all J_{CS} models.*

Proof We show the contrapositive. Suppose X is not valid. Then $\{1 \neg X\}$ is a satisfiable set. The tableau proof construction begins with a tableau having one branch, that containing one formula, $1 \neg X$, and hence begins with a satisfiable tableau. From this only satisfiable tableaus can result, and none of them can be closed. Consequently X is not provable. ■

Completeness can be shown by making use of a systematic tableau construction algorithm or perhaps more simply, a maximal consistent set construction will do. Throughout, assume we use tableaus in conjunction with a constant specification *CS*.

In proving completeness we will consider tableaus that start with an arbitrary finite set of prefixed formulas. Call a set S of prefixed formulas *inconsistent* if some tableau that begins with a finite subset of S closes, following the tableau rules of Section 11. *We assume closure is generalized-atomic closure (or else because of the constant specification) and the single-use restrictions mentioned earlier are followed.* It follows trivially that any superset of an inconsistent set is also inconsistent, something that is used without mention many times below. Call a set *consistent* if it is not inconsistent. As usual, a consistent set S is *maximal* if S cannot be enlarged without becoming inconsistent.

Call a set S of prefixed formulas π-*complete* provided, if $\sigma \pi \in S$ then for some integer k, $\sigma.k \pi_0 \in S$.

Theorem 12.4 (Lindenbaum-Henkin) *Any finite set S of prefixed formulas that is consistent can be extended to a maximally consistent set that is π-complete.*

Proof Let S be a finite set of prefixed formulas that is consistent. The set of all prefixed formulas

is countable—enumerate them as $\sigma_1 X_1$, $\sigma_2 X_2$, ..., and define the following sequence of sets.

$$S_1 = S$$

$$S_{n+1} = \begin{cases} S_n \cup \{\sigma_n X_n\} & \text{if } S_n \cup \{\sigma_n X_n\} \text{ is consistent} \\ & \text{and } X_n \text{ is not } \pi \\ S_n \cup \{\sigma_n X_n, \sigma_n.k\,\pi_0\} & \text{if } S_n \cup \{\sigma_n X_n\} \text{ is consistent,} \\ & X_n \text{ is } \pi, \text{ and } \sigma_n.k \text{ is new} \\ S_n & \text{otherwise} \end{cases}$$

In the second case of this construction one integer k is chosen so that $\sigma_n.k$ does not occur in S_n, and $\sigma_n.k\,\pi_0$ is also added.

Since $S = S_1$ is finite, it follows that every S_n is finite. Then in the second case, where X_n is a π formula, there must be some new prefix $\sigma_n.k$ available, so this case can always be carried out when appropriate.

In cases 1 and 3 it is obvious that S_{n+1} is consistent if S_n is, but this is not immediate for case 2, essentially because of the single-use restrictions. We now establish that case 2 also preserves consistency.

Suppose $S_n \cup \{\sigma_n\,\pi, \sigma_n.k\,\pi_0\}$ is not consistent, where $\sigma_n.k$ is not in S_n. We show $S_n \cup \{\sigma_n\,\pi\}$ is also not consistent, so case 2 does not apply. By assumption there is a generalized-atomic closed tableau for $S_n \cup \{\sigma_n\,\pi, \sigma_n.k\,\pi_0\}$, meeting the single-use restrictions. In this tableau suppose the Prefix Tableau π Rule has been applied to $\sigma_n\,\pi$, say adding $\sigma_n.m\,\pi_0$ to a branch, where $\sigma_n.m$ was new to the branch when it was used. In the node labeled $\sigma_n.m\,\pi_0$ and in its descendants, replace all occurrences of $\sigma_n.m$ with occurrences of $\sigma_n.k$ (thus adding a redundant occurrence of $\sigma_n.k\,\pi_0$ which can be eliminated). One can check that this substitution yields a correctly constructed, generalized-atomic closed tableau. By repeating this process we can eliminate all applications of the π rule to $\sigma_n\,\pi$ and still have a closed tableau, let us call it $\mathcal{T}$. Now start a new tableau, but for the smaller set $S_n \cup \{\sigma_n\,\pi\}$. Begin with an application of the π rule introducing $\sigma_n.k\,\pi_0$, which is allowed because $\sigma_n.k$ is new. Then continue the tableau construction exactly as it was carried out in $\mathcal{T}$. Since $\mathcal{T}$ had no applications of the π rule to $\sigma_n\,\pi$, the new tableau will have a single such application, and it also meets all other single-use restrictions, since $\mathcal{T}$ did, and it is generalized-atomic closed. It follows that $S_n \cup \{\sigma_n\,\pi\}$ is not consistent.

Since S_1 is consistent, it follows that every S_n is consistent too. Let $S_\infty = S_1 \cup S_2 \cup \dots$. It follows that S_∞ is also consistent. It is straightforward to check that S_∞ is maximally consistent, and is π-complete. ∎

We now have 'one-directional' closure results, unlike with axiom system completeness proofs, where the items in the theorem below would be equivalences.

Theorem 12.5 *Let M be a maximally consistent, π-complete set of prefixed formulas. Then all of the following hold.*

1. $\sigma \neg\neg X \in M \Longrightarrow \sigma X \in M$

2. $\sigma\,\alpha \in M \Longrightarrow \sigma\,\alpha_1 \in M$ *and* $\sigma\,\alpha_2 \in M$

3. $\sigma\,\beta \in M \Longrightarrow \sigma\,\beta_1 \in M$ *or* $\sigma\,\beta_2 \in M$

4. $\sigma\,\nu \in M \Longrightarrow \sigma.n\,\nu_0 \in M$ *for every prefix $\sigma.n$ that occurs in M*

5. $\sigma\,\pi \in M \Longrightarrow \sigma.n\,\pi_0 \in M$ *for some prefix $\sigma.n$*

6. $\sigma \neg(t \cdot s){:}Y \in M \implies \sigma \neg t{:}(X \supset Y) \in M$ *or* $\sigma \neg s{:}X \in M$, *for every formula* X

7. $\sigma \neg(s + t){:}X \in M \implies \sigma \neg s{:}X \in M$

8. $\sigma \neg(s + t){:}X \in M \implies \sigma \neg t{:}X \in M$

9. $\sigma\, t{:}X \in M \implies \sigma.n\, X \in M$ *for every prefix* $\sigma.n$ *that occurs in* M

Proof We only check a few items—the rest are similar. Suppose M is maximally consistent and π-complete.

Item 2. Suppose $\sigma\,\alpha \in M$. If we show $M \cup \{\sigma\,\alpha_1, \sigma\,\alpha_2\}$ is consistent, it will follow by maximality that $\sigma\,\alpha_1 \in M$ and $\sigma\,\alpha_2 \in M$. So, suppose $M \cup \{\sigma\,\alpha_1, \sigma\,\alpha_2\}$ is not consistent; we derive a contradiction. By our assumptions, there is some finite subset $M_0 \subseteq M$ such that $M_0 \cup \{\sigma\,\alpha, \sigma\,\alpha_1, \sigma\,\alpha_2\}$ is not consistent, and hence there is a generalized-atomically closed tableau for the set. In this, an application of the Prefix Tableau α Rule to $\sigma\,\alpha$ would add formulas already present, hence is redundant, and we can eliminate that application. Assume this elimination has been done, and hence there is a generalized-atomically closed tableau, call it $\mathcal{T}_1$, in which no rule is applied to $\sigma\,\alpha$. Now, we construct a closed tableau for just $M_0 \cup \{\sigma\,\alpha\}$ as follows. Begin with the members of $M_0 \cup \{\sigma\,\alpha\}$; use the α rule to add $\sigma\,\alpha_1$ and $\sigma\,\alpha_2$; then continue exactly as in $\mathcal{T}_1$. Note that in this new tableau, $\sigma\,\alpha$ has had a rule applied to it just once, meeting the single-use restriction. It follows that $M_0 \cup \{\sigma\,\alpha\}$ is not consistent, but this is a subset of M, which is consistent, and we have a contradiction.

Item 4. Suppose $\sigma\,\nu \in M$, the prefix $\sigma.n$ occurs in M, but $\sigma.n\,\nu_0 \notin M$; we derive a contradiction. As in the previous case, since $\sigma.n\,\nu_0 \notin M$ there must be a finite subset $M_0 \subseteq M$ such that $M_0 \cup \{\sigma.n\,\nu_0\}$ is inconsistent. Since $\sigma.n$ occurs in M there must be some prefixed formula, $\sigma'\,X$, whose prefix σ' begins with (or is) $\sigma.n$. Then $M_0 \cup \{\sigma'\,X, \sigma\,\nu, \sigma.n\,\nu_0\}$ is also not consistent, and hence there is a closed tableau for it; call the tableau $\mathcal{T}_2$. We produce a closed tableau for $M_0 \cup \{\sigma'\,X, \sigma\,\nu\}$. Very simply, begin with the members of $M_0 \cup \{\sigma'\,X, \sigma\,\nu\}$, apply the ν rule to add $\sigma.n\,\nu_0$ to the branch, then proceed as in tableau $\mathcal{T}_2$. Thus $M_0 \cup \{\sigma'\,X, \sigma\,\nu\}$ is inconsistent, but it is a subset of M, contradicting consistency of M.

Item 5. Immediate by π-completeness. ∎

A maximally consistent, π-complete set of prefixed formulas can be used to construct a kind of canonical model. We say how, and then verify its essential properties.

Canonical Model Construction: Let M be a maximally consistent, π-complete set of prefixed formulas. We create a model, $\mathcal{C}(M) = \langle \mathcal{G}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$, called the *canonical model based on* M, as follows.

1. $\mathcal{G}$ is the set of all prefixes that occur in M.

2. If σ and $\sigma.n$ are in $\mathcal{G}$, set $\sigma \mathcal{R} \sigma.n$.

3. If $\sigma \neg t{:}X \notin M$, set $\sigma \in \mathcal{E}(t, X)$.

4. If $\sigma\,P \in M$ for a propositional letter P, set $\sigma \in \mathcal{V}(P)$.

Note the 'double negation' form of item 3 above. If we had a tableau cut rule, we could conclude from $\sigma \neg t{:}X \notin M$ that $\sigma\,t{:}X \in M$, which is more-or-less the condition that is used in axiomatic completeness proofs for justification logics. We don't have cut, but fortunately the more convoluted version stated above still works.

We must verify that the construction of $\mathcal{C}(M)$ produces a J_{CS} model, as defined in Section 10. We only check one of the conditions for $\mathcal{E}$ and leave the rest to the reader. Suppose $\sigma \in \mathcal{E}(s, X)$. Then $\sigma \, \neg s{:}X \notin M$. By Lemma 12.5 part 7, $\sigma \, \neg(s+t){:}X \notin M$, so $\sigma \in \mathcal{E}(s+t, X)$.

We still must check that since tableau proofs assume constant specification CS, then the canonical model meets this constant specification. Constant specifications only provide constants for valid formulas, so suppose A is valid, and hence true at every $\sigma \in \mathcal{G}$ of the canonical model since this, in fact, is a model. Suppose $e_1{:}A \in CS$. Then a tableau branch closes if it contains $\sigma \, \neg e_1{:}A$. It follows that $\sigma \, \neg e_1{:}A \notin M$ for every prefix σ, and hence $\sigma \in \mathcal{E}(e_1, A)$ for every $\sigma \in \mathcal{G}$. It is now easy to see that $\mathcal{M}, \sigma \Vdash e_1{:}A$ for all $\sigma \in \mathcal{G}$ as well. We have just verified that $e_1{:}A$ is true at every possible world of the canonical model under the assumption that $e_1{:}A \in CS$. Next, suppose $e_2{:}e_1{:}A \in CS$. As before, a tableau branch will close if it contains $\sigma \, \neg e_2{:}e_1{:}A$, so $\sigma \, \neg e_2{:}e_1{:}A \notin M$, and hence $\sigma \in \mathcal{E}(e_2, e_1{:}A)$ for every $\sigma \in \mathcal{G}$. It is assumed that CS contains all intermediate specifications, so $e_1{:}A \in CS$ and hence $e_1{:}A$ is true at every possible world, by what was just shown. It follows that $e_2{:}e_1{:}A$ is true at every possible world of the canonical model. And we can continue this argument to sequences of justification constants of any length.

Axiomatically the 'Truth Lemma' involves if-and-only-if conditions, but with tableaus things became one-directional. Again this is because of the absence of a cut rule. Nonetheless, this is still enough for our purposes.

Theorem 12.6 (Truth Lemma) *Let M be a maximally consistent, π-complete set of prefixed formulas, and let $\mathcal{C}(M) = \langle \mathcal{G}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$ be the canonical model based on it. For any $\sigma \in \mathcal{G}$ and formula X:*

$$\sigma \, X \in M \Longrightarrow \mathcal{C}(M), \sigma \Vdash X \tag{10}$$

$$\sigma \, \neg X \in M \Longrightarrow \mathcal{C}(M), \sigma \not\Vdash X \tag{11}$$

Proof The argument is by induction on the complexity of X.

Suppose X is a propositional letter, say P. Condition (10) is by item 4 of the definition of canonical model. For (11), if $\sigma \, \neg P \in M$ then $\sigma \, P \notin M$ since M is consistent. Then by item 4 of the definition again, $\sigma \notin \mathcal{V}(P)$ and so $\mathcal{C}(M), \sigma \not\Vdash P$.

Suppose X is $Y \wedge Z$ and the result is known for Y and for Z. Then, using item 2 of Lemma 12.5,

$$\sigma \, Y \wedge Z \in M \Longrightarrow \sigma \, Y \in M \text{ and } \sigma \, Z \in M$$
$$\Longrightarrow \mathcal{C}(M), \sigma \Vdash Y \text{ and } \mathcal{C}(M), \sigma \Vdash Z$$
$$\Longrightarrow \mathcal{C}(M), \sigma \Vdash Y \wedge Z$$

The other propositional cases, and the modal cases, are similar and are omitted. We discuss the justification term cases in more detail.

Suppose X is $t{:}Y$ and the result is known for Y. We begin with the simpler of the two subcases.

Assume $\sigma \, \neg t{:}Y \in M$. Then by item 3 of the Canonical Model Construction, $\sigma \notin \mathcal{E}(t, Y)$. But then $\mathcal{C}(M), \sigma \not\Vdash t{:}Y$.

Now suppose $\sigma \, t{:}Y \in M$. By item 9 of Lemma 12.5, $\sigma.n \, Y \in M$ for every $\sigma.n \in \mathcal{G}$. By the induction hypothesis, $\mathcal{C}(M), \sigma.n \Vdash Y$ for every $\sigma.n \in \mathcal{G}$. But also, since M is consistent, $\sigma \, \neg t{:}Y \notin M$, and so $\sigma \in \mathcal{E}(t, y)$. Then, using the definition of $\mathcal{R}$ in the canonical model, $\mathcal{C}(M), \sigma \Vdash t{:}Y$. ∎

Now completeness is immediate. Suppose X is not tableau provable. Then there is no closed tableau for $1 \, \neg X$, so the set $\{1 \, \neg X\}$ is consistent. Extend it to a maximally consistent, π-complete set M. Use this set to construct the canonical model $\mathcal{C}(M)$. One of the possible worlds of the model will be 1 and by the Truth Lemma, X will be false at world 1, and hence X is not valid.

13   The Logic **S4LP**

With $\mathsf{J}+\mathsf{K}$ having been discussed, extending the work to **S4LP** is relatively straightforward. We sketch things so the reader has some idea of what is involved. The tableau rules we give are a subset of those for **S4LPN**, from [17]. The language has ! added, as discussed in Section 10.

Once again we are using justification logic models, but applying them to the more general family of implicit/explicit formulas. A model is a J_{CS} model $\mathcal{M} = \langle \mathcal{G}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$, as in Section 10, meeting the following additional conditions that take ! into account. These turn the model into an **LP** model.

- $\mathcal{R}$ is reflexive;

- $\mathcal{R}$ is transitive;

- $\mathcal{E}$ satisfies a monotonicity condition with respect to $\mathcal{R}$: if $\Gamma \mathcal{R} \Delta$ and $\Gamma \in \mathcal{E}(t, X)$ then $\Delta \in \mathcal{E}(t, X)$;

- $\mathcal{E}(t, X) \subseteq \mathcal{E}(!t, t{:}X)$.

In these models $t{:}X \supset X$ is valid for all t and X, as is $t{:}X \supset !t{:}t{:}X$. These are explicit counterparts to $\Box X \supset X$ and $\Box X \supset \Box\Box X$, which are also valid.

Tableau rules for **S4LP** are those for the $\mathsf{K}+\mathsf{J}$ system of Section 11, with the following additions.

Prefix Tableau T Rules	$\dfrac{\sigma\,\nu}{\sigma\,\nu_0}$	$\dfrac{\sigma\,t{:}X}{\sigma\,X}$
Prefix Tableau 4 Rules	$\dfrac{\sigma\,\nu}{\sigma.n\,\nu}$ $\sigma.n$ not new	$\dfrac{\sigma\,t{:}X}{\sigma.n\,t{:}X}$ $\sigma.n$ not new
Prefix Tableau ! Rule	$\dfrac{\sigma\,\neg!t{:}t{:}X}{\sigma\,\neg t{:}X}$	
Prefix Tableau t Monotonicity Rule	$\dfrac{\sigma.n\,\neg t{:}X}{\sigma\,\neg t{:}X}$	

Both Prefix Tableau T Rules are single-use, but are not the only ones that apply to the formulas in question, so formulas cannot be 'crossed off' after these rules are applied. Neither of the Prefix Tableau 4 Rules are single-use. The Prefix Tableau ! Rule and the Prefix Tableau t Monotonicity Rule are both single-use, but not the only rules that might apply to the formulas involved.

To show soundness it is enough to show the tableau rules preserve satisfiability, as in Section 12, except that now satisfiability is with respect to **LP** models. For the Prefix Tableau T Rules this follows from the reflexivity condition. For the rest of the rules one needs the other three of the new conditions for **LP** models. We omit details.

For completeness Theorem 12.4 is applied, but of course consistency is modified to use the tableau rules extended with those just introduced. We omit details. Theorem 12.5 continues to hold, with additional cases corresponding to the new tableau rules.

10. $\sigma\,\nu \in M \Longrightarrow \sigma\,\nu_0 \in M$

11. $\sigma\,t{:}X \in M \Longrightarrow \sigma\,X \in M$

12. $\sigma \nu \in M \implies \sigma.n \nu \in M$ for every prefix $\sigma.n$ that occurs in M

13. $\sigma t{:}X \in M \implies \sigma.n t{:}X \in M$ for every prefix $\sigma.n$ that occurs in M

14. $\sigma \neg!t{:}t{:}X \in M \implies \sigma \neg t{:}X \in M$

15. $\sigma.n \neg t{:}X \in M \implies \sigma \neg t{:}X \in M$

Let M be a maximally consistent, π-complete set; a canonical model $\mathcal{C}(M) = \langle \mathcal{G}, \mathcal{R}, \mathcal{E}, \mathcal{V} \rangle$ is constructed, just as in Section 12, but with one modification: set $\sigma \mathcal{R} \sigma'$ provided σ' is an extension of σ, allowing trivial extensions, that is, equality. (Note that the present definition extends the one in Section 12.)

It must be shown that $\mathcal{C}(M)$ is an LP_{CS} model. Most of the conditions are as before. Transitivity and reflexivity of $\mathcal{R}$ is immediate. The condition that $\mathcal{E}(t, X) \subseteq \mathcal{E}(!t, t{:}X)$ is an easy consequence of the definition of $\mathcal{E}$ and item 14 above. We verify the monotonicity condition. Suppose $\sigma \mathcal{R} \sigma'$ and $\sigma \in \mathcal{E}(t, X)$; we show $\sigma' \in \mathcal{E}(t, X)$. Since $\sigma \mathcal{R} \sigma'$, we must have that $\sigma' = \sigma.n_1.n_2.\cdots.n_k$, where k might be 0. Since $\sigma \in \mathcal{E}(t, X)$, we must have that $\sigma \neg t{:}X \notin M$. Then by repeated use of the contrapositive of item 15 above, we must have $\sigma.n_1.n_2.\cdots.n_k \neg t{:}X \notin M$, and hence $\sigma.n_1.n_2.\cdots.n_k \in \mathcal{E}(t, X)$.

Finally we must prove the Truth Lemma, Theorem 12.6, but for the present logic. Almost all of the cases are as before except for the change in the accessibility relation, which is a minor issue. But there are a few cases that must be modified. Suppose items (10) and (11) from Theorem 12.6 are known for Y; they must be shown for $t{:}Y$. Verification of item (11) is exactly as before, so we only discuss (10). So now suppose $\sigma t{:}Y \in M$. Let $\sigma' \in \mathcal{G}$ be an arbitrary possible world such that $\sigma \mathcal{R} \sigma'$. Then $\sigma' = \sigma.n_1.n_2.\cdots.n_k$, where k might be 0. By repeated use of item 13 above, we have $\sigma.n_1.n_2.\cdots.n_k t{:}Y \in M$. Then by item 11 of the same theorem, $\sigma.n_1.n_2.\cdots.n_{k-1}.n_k Y \in M$, that is, $\sigma' Y \in M$. By the induction hypothesis, (10) gives us $\mathcal{C}(M), \sigma' \Vdash Y$, and this for every σ' accessible from σ. Also, as before, since M is consistent, $\sigma \neg t{:}Y \notin M$, and so $\sigma \in \mathcal{E}(t, Y)$. It follows that $\mathcal{C}(M), \sigma \Vdash t{:}Y$.

Now the completeness proof concludes exactly as in Section 12.

14 The Logics **S4LPN** and **S5LPN**

S4LPN, like S4LP, was introduced in [3]. Axiomatically S4LP combines an axiom system for LP, one for S4, and a connecting axiom, $t{:}X \supset \Box X$. It follows easily that $t{:}X \supset \Box t{:}X$. For S4LPN one also adds $\neg t{:}X \supset \Box \neg t{:}X$. Motivation for this axiom can be found in [3]. A prefixed tableau system for S4LPN was given in [17]. We do not discuss this further here because the semantics for S4LPN requires Artemov-Fitting models, while here we have confined things to the simpler class of Fitting models. Instead we introduce a new but closely related logic that we call S5LPN. For this one adds to S4LPN the familiar axiom $\neg \Box X \supset \Box \neg \Box X$, turning the pure modal part of the logic into S5. In this section we briefly discuss S5LPN, primarily to illustrate the techniques peculiar to prefixed tableaus and nested sequents.

Semantically, one addition is made to the conditions for LP models, from Section 13.

- $\mathcal{R}$ is symmetric.

Of course this makes valid all formulas of the form $\neg \Box X \supset \Box \neg \Box X$. But in addition it also makes valid implicit/explicit formulas of the form $\neg t{:}X \supset \Box \neg t{:}X$. For prefixed tableaus, a system for S4LPN was given in [17], and we simply extend it slightly now. Completeness is proved in

$$\cfrac{\cfrac{\cfrac{\neg x{:}F, [\neg F, F] \quad \langle 5 \rangle}{\neg x{:}F, [F] \quad \langle 4 \rangle}}{\neg x{:}F, \Box F \quad \langle 2, 3 \rangle}}{x{:}F \supset \Box F \quad \langle 1 \rangle}$$

Figure 10: $\mathsf{K} + \mathsf{J}$ Nested Sequent Example

the cited paper, with respect to Artemov-Fitting semantics, but almost no changes are needed to work with Fitting semantics. We omit details. Formally, we add to the rules for $\mathsf{S4LP}$, from Section 13, the following. Note that the Prefix Tableau $4r$ Rule is single-use, but might not be the only applicable rule, and the Prefix Tableau t Dual-Monotonicity Rule is not single-use.

Prefix Tableau 4r Rules $\qquad\qquad\qquad\qquad \cfrac{\sigma.n\,\nu}{\sigma\,\nu} \qquad \cfrac{\sigma.n\,t{:}X}{\sigma\,t{:}X}$

Prefix Tableau t Dual-Monotonicity Rule $\quad \cfrac{\sigma\,\neg t{:}X}{\sigma.n\,\neg t{:}X}$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \sigma.n$ not new

15 Nested Sequents for Justification Logics

In Section 5 we showed how to turn tableau rules into nested sequent rules. The same process can be applied to the prefixed tableau rules for justification logics. We omit details, and simply state results.

For $\mathsf{K} + \mathsf{J}$, we need the nested sequent rules for K, from Section 3, except that now axioms can involve generalized-atomic formulas, and not just propositional letters. To these are added the following translates of the prefix tableau rules from Section 11. As usual, $[\cdots]$ represents a non-empty nested sequent.

Nested Sequent Application Rule

$$\cfrac{\Gamma(t{:}(X \supset Y), (t \cdot s){:}Y) \qquad \Gamma(s{:}X, (t \cdot s){:}Y)}{\Gamma((t \cdot s){:}Y)}$$

Nested Sequent Weakening Rules

$$\cfrac{\Gamma(s{:}X, (s + t){:}X)}{\Gamma((s + t){:}X)} \qquad\qquad \cfrac{\Gamma(t{:}X, (s + t){:}X)}{\Gamma((s + t){:}X)}$$

Nested Sequent Connecting Rule

$$\cfrac{\Gamma(\neg t{:}X, [\neg X, \cdots])}{\Gamma(\neg t{:}X, [\cdots])}$$

In Section 11, Figure 8, we gave a prefixed tableau proof of $x{:}F \supset \Box F$. That translates into the nested sequent proof shown in Figure 10. Numbering corresponds to that of the earlier tableau.

Next we add to the $\mathsf{K} + \mathsf{J}$ nested sequent system additional rules to turn it into a system for the justification logic $\mathsf{S4LP}$. These rules are translates of the prefixed tableau rules of Section 13.

Nested Sequent T Rules
$$\frac{\Gamma(\pi_0, \pi)}{\Gamma(\pi)}$$

$$\frac{\Gamma(\neg X, \neg t{:}X)}{\Gamma(\neg t{:}X)}$$

Nested Sequent 4 Rules
$$\frac{\Gamma(\pi, [\pi, \ldots])}{\Gamma(\pi, [\ldots])}$$

$$\frac{\Gamma(\neg t{:}X, [\neg t{:}X, \ldots])}{\Gamma(\neg t{:}X, [\ldots]))}$$

Nested Sequent ! Rule
$$\frac{\Gamma(t{:}X, !t{:}t{:}X)}{\Gamma(!t{:}t{:}X)}$$

Nested Sequent t Monotonicity Rule
$$\frac{\Gamma(t{:}X, [t{:}X, \ldots])}{\Gamma([t{:}X, \ldots])}$$

And finally, nested sequent versions of the rules that must be added to those above, for S5LPN, deriving from the tableau rules of Section 14.

Nested Sequent 4r Rules
$$\frac{\Gamma(\pi, [\pi, \ldots])}{\Gamma([\pi, \ldots])}$$

$$\frac{\Gamma(\neg t{:}X, [\neg t{:}X, \ldots])}{\Gamma([\neg t{:}X, \ldots])}$$

Nested Sequent t Dual-Monotonicity Rule
$$\frac{\Gamma(t{:}X, [t{:}X, \ldots])}{\Gamma(t{:}X, [\ldots])}$$

16 Suggested Future Work

We have argued that nested sequent modal systems and prefixed tableau systems are notational variants of each other. Since prefixed tableaus have had a much longer history, one benefit of this is that a number of interesting nested sequent systems become available for investigation, in particular for investigation using the proof theoretic methods that have been developed for nested sequent calculi. We mention a few examples of special interest.

In [13, Section 9] prefixed tableaus were introduced for multi-modal logics—logics of knowledge with multiple agents. These are essentially straightforward. Prefixes are now made up of sequences of positive integers in which each (except for the initial 1) is tagged with a particular agent identifier. In [5] nested sequent systems were introduced in which boxed subsequents were indexed with names of agents. It is very likely that the prefixed tableau and the nested sequent systems for multiple agents are notational variants along the lines that we have been examining here. The consequences of this should be worth exploring.

Prefixed tableaus as they were originally introduced in [10, 11] were not even partially modular. As was noted earlier, the more modular tableau rules examined here are from [18]. Still, the earlier rules remain of interest, and they too give rise to nested sequent systems. These are open for investigation.

Finally, there is more to the general area of deep reasoning beyond that which is captured by the modal nested sequent calculi discussed here. More complex machinery has been considered. Likewise prefixed tableaus have, in a sense, been subsumed by the labeled systems of [21]. These

replace the syntactic machinery of prefixes with explicit formulas representing accessibility. They are able to handle modal logics for which no prefixed system exists. It is an interesting question as to how labeled systems and deep reasoning systems relate generally. The central issue is the tree structure inherent in both prefixed tableaus and nested sequent systems, but which is generalized to graphs in labeled systems. Exploring the ramifications of this might be an interesting task.

References

[1] S. N. Artemov. Explicit provability and constructive semantics. *Bulletin of Symbolic Logic*, 7(1):1–36, Mar. 2001.

[2] S. N. Artemov and M. C. Fitting. Justification logic. In E. N. Zalta, editor, *Stanford Encyclopedia of Philosophy*, chapter Logic, Justification. Stanford University, http://plato.stanford.edu/, on-line edition, 2010.

[3] S. N. Artemov and E. Nogina. Introducing justification into epistemic logic. *Journal of Logic and Computation*, 15(6):1059–1073, Dec. 2005.

[4] K. Brünnler. Deep sequent systems for modal logic. *Archive for Mathematical Logic*, 48(6):551–577, 2009.

[5] K. Brünnler. *Nested Sequents*. Habilitation thesis, Institut für Informatik und angewandte Mathematik, Universität Bern, 2010.

[6] M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, editors. *Handbook of Tableau Methods*. Kluwer, Dordrecht, 1999.

[7] F. Fitch. Tree proofs in modal logic. *Journal of Symbolic Logic*, 31:152, 1966. (abstract).

[8] M. Fitting. Semantics and tableaus for LPS4. Technical Report TR–2004016, CUNY Ph.D. Program in Computer Science, Oct. 2004.

[9] M. Fitting. The logic of proofs, semantically. *Annals of Pure and Applied Logic*, 132(1):1–25, Feb. 2005.

[10] M. C. Fitting. Tableau methods of proof for modal logics. *Notre Dame Journal of Formal Logic*, 13:237–247, 1972.

[11] M. C. Fitting. *Proof Methods for Modal and Intuitionistic Logics*. D. Reidel Publishing Co., Dordrecht, 1983.

[12] M. C. Fitting. First-order intensional logic. *Annals of Pure and Applied Logic*, 127:171–193, 2004.

[13] M. C. Fitting. Modal proof theory. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *Handbook of Modal Logic*, chapter 2, pages 85–138. Elsevier, 2007.

[14] M. C. Fitting and R. Mendelsohn. *First-Order Modal Logic*. Kluwer, 1998. Paperback, 1999. Errata at *http://comet.lehman.cuny.edu/fitting/errata/errata.html*.

[15] R. Goré. Tableau methods for modal and temporal logics. Pages 297–396 in [6].

[16] G. E. Hughes and M. J. Cresswell. *A New Introduction to Modal Logic*. Routledge, London, 1996.

[17] H. Kurokawa. Tableaux and hypersequents for Justification Logic. In S. N. Artemov and A. Nerode, editors, *Logical Foundations of Computer Science, International Symposium, LFCS 2009, Deerfield Beach, FL, USA, January 3–6, 2009, Proceedings*, volume 5407 of *Lecture Notes in Computer Science*, pages 295–308. Springer, 2009.

[18] F. Massacci. Strongly analytic tableaux for normal modal logics. In A. Bundy, editor, *Proceedings of CADE 12*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 723–737, Berlin, 1994. Springer-Verlag.

[19] F. Massacci. Single step tableaux for modal logics: Computational properties, complexity and methodology. *Journal of Automated Reasoning*, 24:319–364, 2000.

[20] A. Mkrtychev. Models for the logic of proofs. In S. Adian and A. Nerode, editors, *Logical Foundations of Computer Science, 4th International Symposium, LFCS'97, Yaroslavl, Russia, July 6–12, 1997, Proceedings*, volume 1234 of *Lecture Notes in Computer Science*, pages 266–275. Springer, 1997.

[21] S. Negri. Proof analysis in modal logic. *Journal of Philosophical Logic*, 34:507–544, 2005.

[22] B. Renne. Semantic cut-elimination for two explicit modal logics. In J. Huitink and S. Katrenko, editors, *Proceedings of the Eleventh ESSLLI Student Session, 18th European Summer School in Logic, Language and Information (ESSLLI'06)*, pages 148–158, Málaga, Spain, July 31–Aug. 11, 2006. FoLLI.

[23] B. Renne and E. Goris. A cut-free tableau system for S4LP. In progress.

[24] R. M. Smullyan. *First-Order Logic*. Springer-Verlag, Berlin, 1968. Revised Edition, Dover Press, New York, 1994.