November 1985

Computational Aspects of Proofs in Modal Logic

Greg Hager
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-85-55.

Computational Aspects of Proofs in Modal Logic

Abstract

Various modal logics seem well suited for developing models of knowledge, belief, time, change, causality, and other intensional concepts. Most such systems are related to the classical Lewis systems, and thereby have a substantial body of conventional proof theoretical results. However, most of the applied literature examines modal logics from a semantical point of view, rather than through proof theory. It appears arguments for validity are more clearly stated in terms of a semantical explanation, rather than a classical proof-theoretic one. We feel this is due to the inability of classical proof theories to adequately represent intensional aspects of modal semantics. This thesis develops proof theoretical methods which explicitly represent the underlying semantics of the modal formula in the proof. We initially develop a Gentzen style proof system which contains semantic information in the sequents. This system is, in turn, used to develop natural deduction proofs. Another semantic style proof representation, the *modal expansion tree* is developed. This structure can be used to derive either Gentzen style or Natural Deduction proofs. We then explore ways of automatically generating MET proofs, and prove sound and complete heuristics for that procedure. These results can be extended to most propositional system using a Kripke style semantics and a fist order theory of the possible worlds relation. Examples are presented for standard T, *S*4, and *S*5 systems, systems of knowledge and belief, and common knowledge. A computer program which implements the theory is briefly examined in the appendix.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-85-55.

COMPUTATIONAL ASPECTS OF PROOFS IN MODAL LOGIC

Gregory Donald Hager

MS-CIS-85-55

Department Of Computer and Information Science
Moore School
University of Pennsylvania
Philadelphia, PA 19104

November 1985

Acknowledgements: This work was supported in part by NSF-CER/DCR82-19196 A02, NSF/DCR-8410771, Airforce/F49620-85-K-0018 and ARMY DAAG-29-84-K-0061. This material is based upon work supported under a National Science Foundation Graduate Fellowship.

UNIVERSITY OF PENNSYLVANIA
THE MOORE SCHOOL OF ELECTRICAL ENGINEERING
SCHOOL OF ENGINEERING AND APPLIED SCIENCE

COMPUTATIONAL ASPECTS OF PROOFS IN MODAL LOGIC

Gregory Donald Hager

Philadelphia, Pennsylvania
December, 1985

A thesis presented to the Faculty of Engineering and Applied Science of the University of Pennsylvania in partial fulfillment of the requirements for the degree of Master of Science in Engineering for graduate work in Computer and Information Science.

Dr. Dale Miller

Dr. O. Peter Buneman

Abstract

Various modal logics seem well suited for developing models of knowledge, belief, time, change, causality, and other intensional concepts. Most such systems are related to the classical Lewis systems, and thereby have a substantial body of conventional proof theoretical results. However, most the applied literature examines modal logics from a semantical point of view, rather than through proof theory. It appears arguments for validity are more clearly stated in terms of a semantical explanation, rather than a classical proof-theoretic one. We feel this is due to the inability of classical proof theories to adequately represent intensional aspects of modal semantics. This thesis develops proof theoretical methods which explicitly represent the underlying semantics of the modal formula in the proof. We initially develop a Gentzen style proof system which contains semantic information in the sequents. This system is, in turn, used to develop natural deduction proofs. Another semantic style proof representation, the *modal expansion tree* is developed. This structure can be used to derive either Gentzen style or Natural Deduction proofs. We then explore ways of automatically generating MET proofs, and prove sound and complete heuristics for that procedure. These results can be extended to most propositional system using a Kripke style semantics and a first order theory of the possible worlds relation. Examples are presented for standard T, $S4$, and $S5$ systems, systems of knowledge and belief, and common knowledge. A computer program which implements the theory is briefly examined in the appendix.

Acknowledgements[1]

I am most deeply indebted to Dale Miller for his patient tutelage, his advice and his friendship, (not to mention his TEXmacros). His conscien-cousness, support, and insightful comments (usually never listened to till too late) made doing a thesis an interesting learning experience rather than the drudgery it could have become. This document is unimaginably better due to his detailed reading of earlier drafts, and suggestions for organization.

Thanks to Ruzena Bajcsy for the faith, support and freedom she has offered me since I came to Penn. Of course, the whole GraspLab crew gets a vote of thanks for all the social events that kept me entertained during the summer, and not asking (too often) just what this stuff has to do with robotics anyway.

Finally, a special thanks goes to my office mates for tolerating me this last year through prelims and thesis, and my roomate Franc for not complaining when my part of the apartment started looking like the proverbial disaster area. They were the people responsible for reminding me, when I most needed it, that there are other things to life besides school and modal logic.

[1]This work was supported in part by NSF-CER/DCR82-19196 A02, NSF/DCR-8410771, Airforce/F49620-85-K-0018, and ARMY DAAG-29-84-K-0061. This material is based upon work supported under a National Science Foundations Graduate Fellowship. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the National Science Foundation.

Contents

1 Introduction

The field of *Artificial Intelligence* covers a broad interdisciplinary area of research devoted to the goal of realizing, as computational constructions, those behaviors and performances typically associated with human intelligence. As such, it represents the intersection of work done by researchers in the areas of philosophy, computer science, mathematics, and psychology to name a few.

One particular set of people in the AI community have embraced logic as a language which can be used to model the cognitive processes of intelligent agents at some level of description. Initially, propositional and first order logic were used to model small deterministic situations which could be completely described by some set of axioms. However, philosophers are quick to point out that many human behaviors are *intensional*; and that first order languages tend to be ill-suited to expressing intensional concepts. Some more powerful system should be used.

One can consider two ways of extending first order logic to more powerful systems. One can allow variables and substitution to range over predicates and formula, yielding *higher order* logic; or one can enhance the language with more logical operators, yielding *modal logic*. It is to modal logic that most AI researchers turn when attempting to devise formal systems encompassing intensional concepts.

What does it mean for a system of logic to "model" some concept or concepts. At the very least, we would expect that the sentences which are derivable seem to express the proper (intuitive) relationships among the members of the domain of discourse. Alternatively, if we have a sound and complete theory, we can replace "derivable" by "valid". In this case, we have two competing methods of studying a system of logic. We can construct arguments for sentences from axioms by syntactic methods, or examine them *analytically* by semantic methods.

Formal analytic methods date back to the development of sequential methods by Gentzen [6]. In the first order systems, the *extensional* nature of the language makes it possible to develop an analytic proof theory which stands in exact correspondence to the underlying semantics. However, when we turn to modal languages, analytic methods suffer from the *non-extensional* nature of the language. An analytic proof theory strictly couched in a modal language can never express analytic arguments as a first order language would.

The purpose of this thesis is to develop, by modification of the notion of modal proof, an analytic proof theory somewhat after the style of Gentzen [6]. From this logical basis, we develop computational methods of deduction suitable for both interactive and automatic generation of proofs. This allows the use of the computer as a tool for the study of systems of modal logic – a possibility which seems to have been largely ignored to this point.

1.1 Background and Motivation

Are semantical arguments for validity really useful? That is, do we really need explanations (in the form of proofs) which justify the validity of sentences in an analytic way. There is, of course, no firm answer to a question such as this. On the other hand, a cursory look at some of the uses of modal logic in AI may lend weight to the hypothesis that semantical arguments are useful and should be studied.

From an AI perspective, probably two of the major turning points in the philosophy of modal logics were the development of model theory by Kripke[14], and Hintikka's work [10] on knowledge and belief. The former put modal logic on firm mathematical grounds as a type of logistic system. The latter work set forth systems closely related to the classical $S4$ and $S5$ systems with modal operators which could be interpreted as "agent a knows that p" and "agent a believes that p", thus bringing modal systems to the attention of AI researchers.

From a philosophical point of view, there are several shortcomings with that particular formulation of knowledge and belief. However, it was a starting point for later work by Sato[28]. Based on discussions with John McCarthy, he formalized various extensions to classical modal systems to formalize concepts of knowledge, introspection, and common knowledge [28] as problems in AI. As an example, he showed how the problem of three wise men could be presented using modal logic. It is interesting to note that he also develops a sequent calculus for his modal systems. However, the proof of the wisemen problem is presented *semantically*; and, the formal proof is presented almost as an afterthought.

At about this same time, McCarthy and Hayes[17], in a series of articles on formalized action paradigms, suggested it might be possible to use modal logics (in particular a semantic interpretation of them), as a language for expressing action. Moore [22] did just this by combining both the modal logic of knowledge based on the possible world semantics of Kripke [14], and the situation calculus of McCarthy and Hayes recast in terms of possible worlds. His approach uses a first-order formulation of modal logic semantics, thus allowing explicit reasoning about worlds, within the formalism.

Some of the latest work in modal logic and AI is due to Konolige [13]. He formalizes a *deduction model of belief* for a family of first-order modal logics and proves a form of Herbrand's theorem for these logics. This is crucially important since, while decision procedures have existed for T, $S4$, and $S5$ [11], there have been no corresponding semi-decision procedures for quantified modal logics which are amenable to automated theorem proving.

Another fertile area for the use of modal systems has been in temporal reasoning[16]. Typically, futures and pasts are represented by linear or branching sets of states, and the set of modal operators enhanced to cover all the possibilities of future and past tenses. Thus, a modality can be interpreted, for instance, as saying that "some condition holds in all states of some future" – essentially quantifying existentially on branches, and universally on states within that branch. It seems that semantic proofs in these logics offer a very coherent basis for natural language explanations of temporal system behavior.

One theme which is common to this (quick and incomplete) review of literature is the semantic viewpoint taken toward logic. One would expect, given the computational applications, that the work would have a more computational flavor; that is, manipulation of syntax in a proof theoretical fashion. However, there are few cases where classical modal proofs are even discussed. For instance, Moore circumvents the problem by translating to the first order domain, and Konolige presents an entirely different semantics which is more amenable to computational methods.

There is, in fact, a substantial body of work in the proof theory of modal logics [5,25,26,27,28,29], but, with the notable exception of tableau methods, little of that work has impinged upon current research in applications of the logics. I would suggest the reason why rests fundamentally on the fact that modal logics have an *intensional* interpretation of predicates. Rather than being only strictly true or false, predicates can occur within modalities and be subject to varying interpretations. To accomplish this, the semantics of the modal operators is formalized using items not expressible in the language. This is, of course, where modal languages derive their unique characteristics. It is possible for the modal operators to express fairly abstract concepts and rely on the underlying semantics to supply proper interpretation.

On the other hand, the gap between the syntax and semantics tends to make purely axiomatic arguments for validity somewhat obscure. This is one reason why, even with the availability of abundant proof theory for modal logics, one more often sees semantical arguments for validity offered. e.g. [22,28]. Another problem with strictly axiomatic and in general most proof theoretic systems is that it is difficult to compare modal systems; even those with the same syntax. McCarthy and Hayes [17] make this point, and suggest that systems which explicitly account for the semantics of the logic allow both classification of various systems and "intelligible interpretation for modal predicate calculi."

As mentioned above, very little has been done in the realm of applying computational methods to modal logic. Contrast this with first order languages where the computer is an invaluable aid in generating proofs and refutations. Resolution methods have provided a fertile ground for implementations which are based on logical systems. The fundamental basis for most of this work is some form of *Herbrand's Theorem.*

One of the problems with applying standard proof theoretical tools to theorem proving in modal logic has been the lack of a Herbrand type of result (with the previously cited exception of Konolige). Herbrand's theorem fundamentally rests on the notion of substitution of elements from the domain of discourse yielding a tautologous form of the formula. However, since the underlying semantics of modal systems is nontruth-functional, it is difficult to derive an analog to Herbrand's Theorem. Hence, most automated theorem proving techniques cannot be used in a straightforward fashion.

One way to circumvent this problem is to consider translating modal statements into some first order language and carrying out the proof in that language. Morgan [24] discusses two approaches for applying standard theorem proving technology to non-classical (specifically modal and intuitionistic) logics. The first approach involves embedding the entire axiomatic system of the object language into a meta-language based on first order logic. Axioms and inference rules of the object language appear explicitly in the meta language. This allows for easy translation of proofs of object language statements in the meta language to proofs in the object language. No knowledge of the semantics of the object language is required.

This approach suffers from extreme inefficiency from a computational point of view. We are essentially running an interpreter for one nondeterministic system using another nondeterministic (and perhaps undecidable) system! Last and most important, the approach works only for *propositional* logics; the extensions to first order logic are far from trivial.

The second approach (similar to the one we are taking) translates the formula into first order logic and proceeds with the proof in that system. Morgan [24] and Haspel [9] develop and prove the correctness of proof methods based on a semantic translation of modal statements into first order predicate calculus. Based on this, we know that first order languages are strong enough to express semantical proofs of most modal logics.

The interested reader may wish to examine a version of this approach presented in Moore[22,23]. This method essentially builds an "interpreter" for modal logic in first order logic. A number of axioms are specified for the translation of modal statements into their first order equivalents, and an axiomitization of the semantics of modal logic is used to derive conclusions. He notes the translation axioms can be procedurally interpreted as they are only syntactic rewrites, and proofs which outline a semantical argument for truth are presented. The advantage of this approach is that the reasoning about worlds is very explicit, and it allows a fairly intuitive coalescing of knowledge and action by referring to "states" of knowledge and taking a state space approach to action. [2].

[2]Moore's formulation also deals with identity and quantification which makes the system much

In summary, it appears there are at least two reasons to undertake the study of a form of proof theory which explicitly accounts for the semantics underlying the validity of a statement. First, the best explanations for statements in a theory are often semantical. Secondly, it is conceivable that a proof theory of this form would be more amenable to implementation on a computer.

The theory presented herein addresses both of these issues. It is proof theoretical in that it is applicable to the syntax of the logic; but, it has more correspondence to semantics in that the proofs mirror semantic arguments by showing the actual model structure being developed. In this thesis we develop both the proof theory, and computational methods of deduction for a set of such a systems. The languages we cover will all be propositional modal logics with Kripke style semantics. The restriction to the propositional case is due to merely time and space constraints is not indicative of any known inherent limitations of the method.

1.2 Overview of the Thesis

In the next section of this thesis we review the standard definitions for modal logic. We then provide a definition of the syntax in which proof statements will be written, and supply an interpretation for those formulas. Since we will study a variety of logics in this thesis, we present the syntax and semantics in a general fashion and customize the language and interpretations to each of our examples as needed.

The third section presents a modified sequent calculus for constructing semantical proofs in modal logics having a Kripke type semantics. In the interests of extensibility, the system is presented as a base collection, and operational inference figures for the additional modal operators are added as needed. Information about the possible worlds relation is explicitly represented in the sequents, and the introduction rules for the modalities modify this information. Proofs of soundness and completeness are given for the standard T, $S4$, and $S5$ logics. The system is extended to handle multiple knowers and common knowledge. The Wise Man puzzle as it appears in [28] is presented as a example of the style of proofs using

more complicated. In the propositional case, his approach and ours are essentially the same, though our goals are somewhat divergent.

this formalization.

The next section presents linear natural deduction proofs after the fashion of Miller [18,19,20]. These proofs are much more amenable to machine implementation than the Gentzen proofs upon which they are based, yet capture the same semantic flavor as the sequent calculus. They can be viewed as Suppes style proofs which are "linearization" of the sequent system established in the previous section. The presentation will be based on a proof *outline*. Several outline transformations will process outlines until a completed proof is achieved. Correctness of these transformations will be based on the Gentzen system. As a result, the proof of correctness gives a simple algorithm for constructing a Gentzen proof corresponding to the natural deduction proof.

Following this, we develop the notion of *proof representations* for modal statements. In particular, we adopt the *expansion tree* as a basic proof representation and modify it to suit our languages. Expansion trees and their modal analogs have many desirable properties as a compact and clean proof representation. As an example, we will demonstrate how sequential and natural deduction proofs can be constructed from expansion tree proofs.

One can view the search for an expansion tree proof in two stages. First, construct the "scaffolding" of the tree, then search for the appropriate substitution instances of a formula which make the deep structure of the tree tautologous. One way of searching for these substitution instances is to use the method of matings [1,2]. We will examine some of the issues involved in this proof process. In particular, we will show that substantial speedup can be achieved by suitably structuring the search. These results also suggest that it is possible to view the search for a proof as interaction between a standard theorem prover, and attached procedures. There are several issues of control in this view; with standard tableau systems appearing as a point on the spectrum.

Finally, we will describe implemented systems for finding expansion trees, and editing natural deduction proofs based on the above theory. Nearly all examples presented in the thesis were run on or generated by these systems. In the appendix we present some Prolog code corresponding to the mathematical definitions in order to clarify the connections between the formalism and its implementation.

It is assumed the reader is familiar with standard first order logic, and has had some exposure to modal logic and higher order logic. Knowledge of Gentzen systems and natural deduction proofs is essential, as well as a basic understanding of automated theorem proving. Familiarity with the language Prolog is useful, but not essential.

2 Syntax and Semantics

This section gives a general account of some modal systems of logic – that is, logics with an extended set of operators. Since we will study a number of languages, our definitions of syntax will be for languages with *some set* of modal operators. Similarly, our definitions of semantics will be general enough to account for this class of languages.

We will in fact supply two definitions of syntax. The first is the standard definition of a modal language. This, in our case, is what might be termed the *outer* syntax. It is the syntax of the language under study. On the other hand, our proof system will have a special syntax, the *inner* syntax – the language in which proofs are carried out. Our definition of "proof" in the next section will make the connection between statements of the outer language, and proofs of those statments couched in the inner language.

2.1 Definition of the Outer Language

The syntax for modal languages is the normal set of well formed formulae of the propositional calculus closed under both the standard rules of formation, and rules for the modal operator symbols. Following [4], we will adopt the convention of referring to some sequence of modalities and negations containing an even number of negations as *affirmative,* and an odd number as *negative.* Often, a modal language contains modalities which are *dual* to each other – that is, either operator can be defined as an affirmative occurence of the other. In order to generalize the presentation, we will consider the set $\mathcal{R}$ to be some set of modal operator symbols, and some subset of their duals, Ω, will be introduced by definition if they are required.

In the sequel, we will take the liberty of using the symbol $\curlywedge\!\!\!\curlyvee$ as a schema to be uniformly replaced by $\wedge$ or $\vee$ wherever it occurs; and, similarly, $\mathcal{Q}$ will stand for $\forall$ or $\exists$ whichever being appropriate from context. Also, more recent literature has replaced L and M by the symbols $\Box$ and $\Diamond$ respectively. In this thesis we will continue to use L and M with their classical interpretations. Lastly, we will let characters at the beginning of the greek alphabet (α, $\beta, \ldots$) denote formulas in the outer syntax, and characters toward the end (ψ, $\sigma, \ldots$) as formulas in the inner syntax.

Definition 2.1 Let Π be a set of propositional atoms, and $\mathcal{R}$ be a (possibly empty) set of modal operators. $\mathcal{P}$, the set of well-formed propositions, is the smallest set containing Π and closed under the following rules of formation:

- if α is a wff, then $\sim \alpha$ is a wff
- if α and β are a wffs, then $\alpha \vee \beta$ is a wff
- if α is a wff, then $\rho\alpha$ is a wff for all $\rho \in \mathcal{R}$.

We will often enrich our syntax by defining the symbol $\supset$ and a set of symbols, Ω, which represent the duals of the operators in $\mathcal{R}$.

Definition 2.2

- $\alpha \supset \beta \stackrel{df}{\equiv} \sim \alpha \vee \beta$
- For each $\omega \in \Omega$, $\omega\alpha \stackrel{df}{\equiv} \sim \rho \sim \alpha$ for some unique $\rho \in \mathcal{R}$.

In this formulation, we can define the system for a single modal operator by letting $\mathcal{R} = \{L\}$ and $\Omega = \emptyset$, or get standard syntax for T ($S4$ or $S5$) by adding M to Ω. The systems of knowledge [8,13,22] consider a number of modalities, K_i or S_i representing the knowledge or beliefs of agents in the system. We can arrive at modal logic of knowledge by letting $\mathcal{R} = \{K_1, \ldots, K_n\}$ and $\Omega = \emptyset$

Now that we have a syntax, we need to supply an interpretation to give content to the system. There is a fairly uniform semantics which, with only minor variations, covers all systems of modal logic we will discuss. The formulation we will present initially is the standard account such as is found in [4,11,12]. This is not the only possible semantical account for modal logics. For instance, Halpern [7] has given an alternate semantics which is also sound and complete with respect to a modal language used to model knowledge, and Konolige[13] formalizes yet another interpretation which is equivalent to Kripke semantics under certain restrictions. At this point, however, we will consider only Kripke type semantics.

The basic notion in Kripke semantics is a world (sometimes referred to as a state in the propositional case). A world is an assignment of truth valuation to atomic symbols, or alternatively a set of propositions. Thus, the semantics of propositions under modal operators is not directly truth-functional – a statement in the language may be true or false depending on the particular world we choose to look at.

Worlds are related by an *accessibility relation*. This accessibility relation is used to give an interpretation for the modal operators. The interpretation of a modality can often be thought of as *quantifying* universally or existentially over related worlds. We will refer to modalities with the former interpretation as *universal* modalities, and the latter as *existential* modalities.

Definition 2.3 A model, M, is a tuple $(W, V, R_{\rho_1}, \ldots, R_{\rho_n})$ where

- W is a set of worlds,
- V is a valuation function $V : \Pi \times W \mapsto \{0,1\}$, and
- R_{ρ_i} for any $\rho_i \in \mathcal{R}$ is a binary relation on worlds in W, i.e. $R_{\rho_i} \subseteq W \times W$.

Given this structure, we can define an interpretation for a modal formula.

Definition 2.4 A formula, α, is *true* in a model, M, and a world, $w_i \in W$ (denoted $\langle M, w_i \rangle \models \alpha$) iff:

- $\langle M, w_i \rangle \models p$ iff $V(p, w_i) = 1$ if $p \in \Pi$,
- $\langle M, w_i \rangle \models \sim \alpha$ iff not $\langle M, w_i \rangle \models \alpha$
- $\langle M, w_i \rangle \models \alpha \vee \beta$ iff $\langle M, w_i \rangle \models \alpha$ or $\langle M, w_i \rangle \models \beta$
- $\langle M, w_i \rangle \models \alpha \wedge \beta$ iff $\langle M, w_i \rangle \models \alpha$ and $\langle M, w_i \rangle \models \beta$
- for any universal $\rho \in \mathcal{R}$, $\langle M, w_i \rangle \models \rho\alpha$ iff for all w_j such that $w_i R_\rho w_j$, $\langle M, w_j \rangle \models \alpha$
- for any existential $\rho \in \mathcal{R}$, $\langle M, w_i \rangle \models \rho\alpha$ iff there is some w_j such that $w_i R_\rho w_j$, and $\langle M, w_j \rangle \models \alpha$

Definition 2.5 A formula, α, is *true in a model* (denoted $M \models \alpha$) iff it is true at every world in that model.

Definition 2.6 A formula α is x-valid for some system of modal logic x (denoted $\models_x \alpha$) iff it is true in all models of the class conforming to the restrictions of system x.

Different systems of modal logic can be defined over the same language by modifying the class of models under consideration. For instance, having no restrictions yields a basic system (sometimes called K, not to be confused with the system K in the thesis) which is the smallest system one can consider. Any system which contains K is called a ***normal*** system. T, $S4$, and $S5$ are normal systems which restrict the relation, R_L, be reflexive, reflexive and transitive, or an equivalence respectively. [3] All of the system we will present are normal systems.

2.2 Definition of the Inner Language

In the previous subsection, we defined validity, in standard fashion, as satisfaction ranging over all models and all worlds in that model. This is not the only possible definition. In his original work, Kripke defined a ***normal model*** to be a triple $\langle H, W, R\rangle$ where W is a set of worlds, R is a relation, and H is some arbitrary member of W. The interpretation function then uses H as the ***originating*** member in the world relation.

As one would expect, Kripke's definition of validity implies the one we have given above. If a formula is satisfied in a model from some arbitrary world, then by universal generalization, it is true in all worlds. Notice, then, that the truth of some formula in a world depends only on the subformula of that formula, and the worlds related to that originating world. This is formally expressed by defining the notion of a ***generated*** model[4]. We say a model M^ω is ***generated*** from M by ω if M^ω is a restriction of M to the worlds related to ω. The particular result we are interested in is:

Proposition 2.1 Let M be a model. Then:

$$M \models \mathcal{A} \text{ iff for every } \omega \in M, M^\omega \models \mathcal{A}$$

In other words, the generated portion of a model suffices as a test of the truth of a formula in a model. For purposes of validity, then, we need only focus our

[3]Nearly all work in AI has been done using variants of the systems mentioned above. There are several other restrictions possible on R (e.g. Euclidean or Serial) which we will neglect. Most results presented in this thesis generalize to those variants.

attention on the class of generated models. [4]

This view of semantics seems to lend itself toward an analytical style of analysis. If one were analyzing the semantical content of a formula, one would pick some arbitrary world from which to start. One would then examine the model generated by the sentence to determine truth or falsity. This is, in fact, the basis for most refutation-based semantic tableau theorem provers.

In order to construct an analytic proof theory in this style, we must be able to represent the intermediate stages of the proof. This is not (directly) possible in the standard syntax because propositions are not strictly *extensional*. Modal languages are not extensional simply because the truth valuation of a proposition depends on both its subformula, *and* the possible world it occurs in, as can be clearly seen in Definition 2.4. We intend to develop a proof structure which is closely related to the semantics of the formulae, so we must tailor that structure to be more nearly truth-functional. Obviously, the way to do this is to develop some way of representing the possible-worlds relation.

In our proof systems, we will ornament formulae with a world term denoting the world or set of worlds in which the subformula is to be interpreted, e.g. $(\alpha \vee \beta)_{w0}$. The proof will proceed by generating a model based on the sentence. At all points, the ornamentation on the formula, and a set of relation constraints will encode the current frame of the proof. Validity will be ensured by selecting a special arbitrary initial world term which appears nowhere else in the proof and appealing to universal generalization over the class of models generated from that term.

The following definitions formalize the notion of ornamented formulas, which we will sometimes call the *inner syntax*. The standard definition of modal formulas will be called the *outer syntax*.

Definition 2.7 Let $\Xi = \{w_0, w_1, \ldots\}$, be a set of *world variables*.

Let Π be a set of propositional atoms, and $\mathcal{R}$ be a (possibly empty) set of modal operators. $\mathcal{P}_I$, the set of well-formed inner propositions, is the smallest set closed under the following rules of formation:

- $\{(\alpha)_w | \alpha \in \Pi \text{ and } w \in \Xi\} \subseteq \mathcal{P}_I$,

[4] For more discussion on generated models, the interested reader should consult [4].

- if $(\alpha)_w$ is a wff, then $(\sim \alpha)_w$ is a wff
- if $(\alpha)_w$ and β_w are a wffs, then $(\alpha \boxtimes \beta)_w$ is a wff
- if $(\alpha)_w$ is a wff, then $(\rho\alpha)_w$ is a wff for all $\rho \in \mathcal{R}$.

This is simply a restatement of Definition 2.1 so that all well-formed formulas are ornamented with a world term. However, here we have an explicit representation for possible worlds. This requires semantic modifications to account for an interpretation of the world variables and the possible worlds relation. We will do this in the obvious way, by interpreting the modal formulae under a substitution of elements of the possible worlds domain for the world variables.

Our proof systems will be sequent systems similar to those developed by [6]. However, we will augment the sequent with a representation for constraints on the relationship among the possible worlds variables. That is, sequents in this system are to be of the form

$$\Re; \Sigma \rightarrow \Theta$$

where Σ and Θ are sets of ornamented modal formulae, and $\Re$ contains statements of the form: $w_i \, R_\rho \, w_j$. These latter statements encode the current conditions on the possible worlds relation for the various modal operators (ρ in this case) in the language. Our notion of substitution will have to account for these constraints.

Definition 2.8 Define a *substitution* as a mapping, s, from elements of Ξ to elements of the possible worlds domain, W, of a model, i.e.

$$s : \Xi \rightarrow W$$

Moreover, we will call a substitution *proper* with respect to some set of statements[5] $\{w_j \, R_{\rho_i} \, w_k ...\}$ if, for all such statements,

$$\langle s(w_j), s(w_k) \rangle \in R_{\rho_i}$$

A proper substitution, then, is a mapping of the possible world ornamentations to the possible worlds of a model such that the relational constraints are met. By using the above definition, we offer the following interpretation for our extended sequents:

[5]Note we denote *syntax* by $w \, R_\rho \, w'$ and write this actual relation in a model as $\langle w, w' \rangle \in R_\rho$ to avoid confusion.

Definition 2.9 Let Γ and Δ be possibly empty sets of inner formulas. We say M is a model for a sequent $\Re; \Gamma \longrightarrow \Delta$, written

$$M \models \Re; \Gamma \longrightarrow \Delta$$

if, for every $\Re$-proper substitution s, we have

$$\langle M, s(w) \rangle \not\models \alpha \text{ for some } (\alpha)_w \in \Gamma \text{ or } \langle M, s(w) \rangle \models \beta \text{ for some } (\beta)_w \in \Delta$$

3 A Gentzen-style System for Modal Logic

In this section we will present several flavors of a sequent system, LKM, for modal logic after the style of Gentzen [6]. In section 2, we introduced a notation which allows us to specify the possible worlds information needed to interpret modal statements. Recall our convention is the expression $(\mathcal{A})_w$ is interpreted to mean: "The formula $\mathcal{A}$ is true in the world denoted by w", and our sequents contained statments which denoted the relationships between possible worlds.

Inference rules in a sequent system are typically taken to be *introduction* rules for each logical connective in the language. In order to generalize our treatment, we will define a base collection of rules for LKM which correspond to the standard complete set of operational inference figures for non-modal calculus. For any modal language with some operator set $\mathcal{R}$, we will add an antecedent and a succedent introduction rule for every $\rho \in \mathcal{R}$. By symmetry, every $\omega \in \Omega$ will use the introduction rules of it's corresponding member in $\mathcal{R}$, but with the side of the occurrence reversed (i.e. the succedent rules become antecedent rules and vice-versa). By adding these inference figures to the base collection to get particularizations of LKM to the syntax under study.

As a proof proceeds, certain introduction rules will modify the "topology" of the possible worlds relation by adding constraints to $\mathfrak{R}$. Other introduction rules will use the information in $\mathfrak{R}$ as a proviso on their applicability. The restrictions of this proviso allow us to specialize a particular instantiation of LKM to some class of models. At the end of the proof, $\mathfrak{R}$ gives the essential structure of the possible worlds relation which forces validity.

In the sequel, we will adopt the convention of denoting different instances of LKM by a subscript naming the logistic system in use, e.g. LKM_{s4}, or LKM_{s5}. The subscripts will be one of T, $S4$, $S5$, K (knowledge), or O (common knowledge). Also, we will denote the derivability of an endsequent of the form $\emptyset; \longrightarrow (\alpha)_w$ in a system x by $\vdash_x \alpha$, and the derivability of a proposition in a Hilbert system by $\vdash_H$. Furthermore, we will use the symbol $\Vdash^x$ to denote derivability in the (usually first order) theory of possible worlds relations for a system of modal logic, x. For example, $\Vdash^{S4}$ denotes derivability in a theory of reflexive, transitive, binary relations.

3.1 The Basic System LKM

Let A, C, and P be propositional formulas, and let Γ, Δ, Θ and Λ represent possibly empty, finite lists of formulae. The following are the base inference rules for our modal derivation system, LKM.

Definition 3.1 (The Base Collection of Inference Figures) The base collection of inference figures is:

$$\frac{\Re;\Gamma \longrightarrow \Theta}{\Re;(A)_w,\Gamma \longrightarrow \Theta}\ \text{thinning} \qquad \frac{\Re;\Gamma \longrightarrow \Theta}{\Re;\Gamma \longrightarrow \Theta,(A)_w}\ \text{thinning}$$

$$\frac{\Re;(A)_w,(A)_w,\Gamma \longrightarrow \Theta}{\Re;(A)_w,\Gamma \longrightarrow \Theta}\ \text{contraction} \qquad \frac{\Re;\Gamma \longrightarrow \Theta,(A)_w,(A)_w}{\Re;\Gamma \longrightarrow \Theta,(A)_w}\ \text{contraction}$$

$$\frac{\Re;\Delta,(A)_{w1},(C)_{w2},\Gamma \longrightarrow \Theta}{\Re;\Delta,(C)_{w2},(A)_{w1},\Gamma \longrightarrow \Theta}\ \text{interchange} \qquad \frac{\Re;\Gamma \longrightarrow \Theta,(A)_{w1},(C)_{w2},\Lambda}{\Re;\Gamma \longrightarrow \Theta,(C)_{w2},(A)_{w1},\Lambda}\ \text{interchange}$$

$$\frac{\Re;\Gamma \longrightarrow \Theta,(A)_w \qquad \Re;(A)_w,\Delta \longrightarrow \Lambda}{\Re;\Gamma,\Delta \longrightarrow \Theta,\Lambda}\ \text{cut}$$

$$\frac{\Re;\Gamma \longrightarrow \Theta,(A)_w \qquad \Re;\Gamma \longrightarrow \Theta,(C)_w}{\Re;\Gamma \longrightarrow \Theta,(A\wedge C)_w}\ \wedge\text{–IS}$$

$$\frac{\Re;(A)_w,\Gamma \longrightarrow \Theta}{\Re;(A\wedge C)_w,\Gamma \longrightarrow \Theta}\ \wedge\text{–IA} \qquad \frac{\Re;(C)_w,\Gamma \longrightarrow \Theta}{\Re;(A\wedge C)_w,\Gamma \longrightarrow \Theta}\ \wedge\text{–IA}$$

$$\frac{\Re;(A)_w,\Gamma \longrightarrow \Theta \qquad \Re;C_w,\Gamma \longrightarrow \Theta}{\Re;(A\vee C)_w,\Gamma \longrightarrow \Theta}\ \vee\text{–IA}$$

$$\frac{\Re;\Gamma \longrightarrow \Theta,(A)_w}{\Re;\Gamma \longrightarrow \Theta,(A\vee C)_w}\ \vee\text{–IS} \qquad \frac{\Re;\Gamma \longrightarrow \Theta,(C)_w}{\Re;\Gamma \longrightarrow \Theta,(A\vee C)_w}\ \vee\text{–IS}$$

$$\frac{\Re;\Gamma \longrightarrow \Theta,(A)_w}{\Re;(\neg A)_w,\Gamma \longrightarrow \Theta}\ \neg\text{–IA} \qquad \frac{\Re;(A)_w,\Gamma \longrightarrow \Theta}{\Re;\Gamma \longrightarrow \Theta,(\neg A)_w}\ \neg\text{–IS}$$

$$\frac{\Re; (A)_w, \Gamma \longrightarrow \Theta, (C)_w}{\Re; \Gamma \longrightarrow \Theta, (A \supset C)_w} \supset\text{-IS}$$

$$\frac{\Re; \Gamma \longrightarrow \Theta, (A)_w \qquad \Re; (C)_w, \Delta \longrightarrow \Lambda}{\Re; (A \supset C)_w, \Gamma, \Delta \longrightarrow \Theta, \Lambda} \supset\text{-IA}$$

Axioms in this system are of the form:

$$\Re; (\mathcal{A})_x \rightarrow (\mathcal{A})_x$$

A proof in this system will be a tree with the sequent $\emptyset; \rightarrow (\mathcal{A})_w$ at the root, and axioms at the leaves. We will say this is a proof *for* the modal statement $\mathcal{A}$.

Proposition 3.1 (Soundness and Completeness of the Base Collection) There is an LKM proof for α if and only if α is tautologous.

Notice that any proof in LK could be converted to a proof in LKM merely by adding a $w \in \Xi$ ornamentation to all formulae in all sequents, and the converse. Hence, LK and LKM proofs are equivalent, and, by soundness and completeness of LK for propositional logic, LKM is also sound and complete.

Corollary 3.2 Propositional calculus is contained in all modal systems discussed in this thesis.

3.2 The Systems T, $S4$, and $S5$

Most classical modal languages have a single modal operator. These are based on a possible worlds relation which can be reflexive, a pre-ordering, or an equivalence relation[6]. These systems are typically called T, $S4$, and $S5$ respectively.

Consider enriching the syntax of our base language with the standard modalities L and M, i.e $\mathcal{R} = \{L\}$ and $\Omega = \{M\}$. We will then need to add inference figures to the base collection to account for these operators. Heretofore, nothing was done with the topology of the possible worlds relation. All generated models contained a single world and an empty relation. By introducing these modalities, we also introduce the need to state and use restrictions on the models generated. Hence, the following inference figures make explicit reference to $\Re$.

[6] It can also be serial or euclidean, or have other constraints. See [8,4].

Definition 3.2

$$\frac{[\Re \Vdash^s w\, R_L\, x] \;\; \Re; (p)_x, \Sigma \longrightarrow \Theta}{\Re; (Lp)_w, \Sigma \longrightarrow \Theta}\; L\text{–IA}^{\dagger} \qquad \frac{\Re, w\, R_L\, x; \Sigma \longrightarrow (p)_x, \Theta}{\Re; \Sigma \longrightarrow (Lp)_w, \Theta}\; L\text{–IS}^{\ddagger}$$

$$\frac{\Re, w\, R_L\, x; p_x, \Sigma \longrightarrow \Theta}{\Re; (Mp)_w, \Sigma \longrightarrow \Theta}\; M\text{–IA}^{\ddagger} \qquad \frac{[\Re \Vdash^s w\, R_L\, x] \;\; \Re; \Sigma \longrightarrow p_x, \Theta}{\Re; \Sigma \longrightarrow (Mp)_w, \Theta}\; M\text{–IS}^{\dagger}$$

†These rules contain the proviso that $w\, R_L\, x$ follows, in the theory of the system s, from the statements contained in $\Re$, i.e $\Re \Vdash^s w\, R_L\, x$ as indicated on the inference figure. Henceforth we will adopt the convention the bracketed figure represents this proviso and refrain from noting it explicitly.

‡These rules contain the proviso that x does not appear as an ornamentation or variable in the lower sequent.

Furthermore, we will add the following constraints to the possible worlds relation. The theory of s will contain:

- The axiom of reflexivity, $\forall x\, .x\, R_L\, x$, if s is T, $S4$, or $S5$;
- The axiom of transitivity, $\forall x\, y\, z\, .x\, R_L\, y \wedge y\, R_L\, z \supset x\, R_L\, z$, if s is $S4$ or $S5$;
- The axiom of symmetry, $\forall x\, y\, .x\, R_L\, y \supset y\, R_L\, x$, if s is $S5$.

We will refer to these particularizations of LKM as LKM_t, LKM_{s4}, and LKM_{s5} respectively.

Example 3.1 A proof of $Lp \supset LLp$ in LKM_{s4} or LKM_{s5} would be:

$$\cfrac{\cfrac{\cfrac{\cfrac{\Re_2; p_{w_2} \longrightarrow p_{w_2}}{\Re_2; Lp_{w_0} \longrightarrow p_{w_2}}\;\text{L-IA}}{\Re_1; Lp_{w_0} \longrightarrow Lp_{w_1}}\;\text{L-IS}}{\emptyset; Lp_{w_0} \longrightarrow LLp_{w_0}}\;\text{L-IS}}{\emptyset; \longrightarrow (Lp \supset LLp)_{w_0}}\;\supset\text{–IA}$$

$\Re_1 = \{w_0\, R_L\, w_1\}$
$\Re_2 = \{w_0\, R_L\, w_1, w_1\, R_L\, w_2\}$

We could prove soundness of these systems based on the validity preserving properties of a first order translation of a modal statement[7]. Instead, we shall offer the following direct proof based on our interpretation for $\mathfrak{R}$.

Proposition 3.3 (Soundness for $LKM_{T,S4,S5}$) If $\vdash_{T,S4,S5} \alpha$ then $\models_{T,S4,S5} \alpha$

> **Proof:** We merely need to show the additional rules are validity preserving. We will demonstrate the proof for the L rules. The proof for M is, by symmetry, a trivial modification.
>
> - Consider L—IS. Assume the upper sequent of the rule is valid, but there is a counter model, M, for the lower one. Then there is a substitution s such that $\langle M, s(w)\rangle \not\models Lp$. But, by the proviso that x does not appear in the lower sequent, and the validity of the upper sequent we know that $\langle M, s(x)\rangle \models p$ for any $\mathfrak{R}$-proper substitution, and hence for all worlds accessible from $s(w)$ in all models, including $s(x)$ in M. This is a contradiction, so there can be no such countermodel.
> - Consider L—IA. Assume the upper sequent is valid, but there is a countermodel, M, to the lower sequent. Then, there is a substitution s such that $\langle M, s(w)\rangle \models Lp$. But, by validity of the upper sequent, we know that $\langle M, s(x)\rangle \not\models p$. By the proviso, it must be the case that $s(w), s(x) \in R_L$. This is a contradiction.
>
> ♣

We will show completeness of this system *relative* to the axiomitization for the Hilbert system given in definition C.1, and then appeal to the completeness of that system. The method of the proof is to show that any formula derived in some Hilbert system has a corresponding proof in the Gentzen system. In this proof, we will make essential use of cut; so, the proof demonstrates completeness for the system *with* the cut rule. We must then demonstrate that *cut-elimination* holds for this system to support a completeness result for the system without cut.

The following lemmas demonstrate that, given an LKM proof for the inputs to necessitation or modus ponens, we can generate a proof for the output. Formally, these rules are *admissible* to the system. [8]

[7] See the appendix for this alternate soundness proof.

[8] A rule is *admissible* to a system if adding it to the system does not change the theorems of the system[13].

Lemma 3.4 The rule of necessitation is admissible to this system.

Proof: If we have a proof for α, we can construct a proof for $L\alpha$ by the following procedure. Select a new world variable w' not appearing anywhere in the proof for α. Add the statement $w' \, R_L \, w$, where w if the ornamentation of α, to $\mathfrak{R}$ of the bottom sequent. Apply L–IS to the bottom sequent, producing a sequent $\emptyset; \longrightarrow (L\alpha)_{w'}$. This, then, is a proof for $L\alpha$.

♣

Lemma 3.5 The rule of modus ponens is admissible to this system.

Proof:

Suppose we have proofs for $(\alpha)_w$ and $(\alpha \supset \beta)'_w$. Then by suitable renaming of ornamentation, we can construct a proof for $(\beta)_w$ in the following fashion. First, note that last step in the proof for $\alpha \supset \beta$ was $\supset$ –IS. Thus the new proof would join at this last step and become:

$$\frac{\emptyset; \longrightarrow (\alpha)_w \qquad\qquad \emptyset; (\alpha)_w \longrightarrow (\beta)_w}{\emptyset; \longrightarrow (\beta)_w} \text{ cut}$$

♣

Proposition 3.6 (Relative Completeness of LKM_{s5}) If $\vdash_H \alpha$ then $\vdash_{T,S4,S5} \alpha$

Proof: The proof is by induction on the length of the Hilbert derivation.

- A derivation of length zero consists of some axiom of the system. It is easy to show that all axioms of the systems under question are derivable in the appropriate instantiation of LKM
- A derivation of length n is some derivation of length $n-1$ followed by the application of one rule of inference to some subset of the formulas appearing in that derivation. By the induction hypothesis, there is an LKM proof for those formulas in the system under question. By the admissibility of the inference rules, we know we can construct a proof for the the final formula in the sequence based on the proofs for the basis of the inference rule.

♣

Corollary 3.7 If $\models_{T,S4,S5} \alpha$ then $\vdash_{T,S4,S5} \alpha$

Such a completeness result for Gentzen systems without cut rests fundamentally on *cut-elimination* for the Gentzen system. We will not prove cut-elimination directly for of our systems at this point, but postpone the argument to Section 5 where it will be a corollary of a proof transformation.

Systems without cut also exhibit an interesting relationship between ornamentations and members of $\Re$ in a sequent. If we examine the modality rules, we see that any world ornament appearing in a sequent *must* also appear in $\Re$ except for the initial world. Also, if any relation constraints appear in $\Re$, then $w0$ appears there also. We will formally call this property the *containment property* .

Definition 3.3 (The Containment Property) A system exhibits the containment property if, for any sequent derivable in the system, either $\Re$ contains no constraints, or it is the case that any world ornament of a formula in the sequent appears in $\Re$ of that sequent.

All systems in this thesis in fact have the containment property. One immediate consequence of the containment property is the sharpening of the provisos requiring arbitrary world variables. Instead of looking at the entire sequent, it now suffices to consider only the members of $\Re$.

Also, notice that it is possible that the model generated by a proof with cut may not be a generated model. In fact, it may not even be a *cohesive* model – that is, a model where the possible worlds topology is connected. Thus, by showing cut-elimination for this system, we also have an alternate proof for proposition 2.1.

The exact relationship between cut and the model topology is an issue that requires further investigation.

3.3 Extensions of LKM to a Logic of Knowledge

The extension of LKM to multiple agents in a system of knowledge is very straightforward. We now assume there are a number of modal operators $\mathcal{R} = \{K_1, K_2, \ldots\}$, and $\Omega = \emptyset$. The rules $L-$IA and $L-$IS become a set of rules K_i-IS and K_i-IA for $i = 1, \ldots$, and the rules $M-$IA and $M-$IS are disallowed. The accessibility relation will be indexed by agent, i.e. there will be n accessibility relations R_{K_i}. The possible worlds theory for K will contain the conjunction of the appropriate constrains on each accessibility relation. The additional rules for LKM_K are (schematically):

$$\frac{[\Re \Vdash^K w\, R_{K_i}\, x] \;\; \Re; (p)_x, \Sigma \longrightarrow \Theta}{\Re; (K_i p)_w, \Sigma \longrightarrow \Theta}\; K_i\text{–IA} \qquad \frac{\Re, w\, R_{K_i}\, x; \Sigma \longrightarrow (p)_x, \Theta}{\Re; \Sigma \longrightarrow (K_i p)_w, \Theta}\; K_i\text{–IS}^{\ddagger}$$

‡This rule contains the proviso that x does not appear as a variable on $\Re$ or an ornamentation of a formula in the lower sequent.

The theory of K's accessibility relation will contain an axiomitization of the possible worlds relation for each "knower". Hence we can represent agents which differing reasoning abilities (e.g. positive introspection, negative introspection) by varying the restrictions on the different accessibility relations. The proofs of soundness and completeness of these rules is similar to the Propositions 3.3 and 3.6, and so are omitted.

Modal logics of knowledge are often enhanced by adding modalities indicating common knowledge (sometimes differentiating between *implicit* common knowledge, and *explicit* common knowledge [28,8]). For instance, the following modifications of LKM_K extend that system to account for a modal operator O representing common knowledge:

Definition 3.4

1. The addition of a modality, O to $\mathcal{R}$.
2. The addition of preorder constraints on R_O to the theory of K.
3. The addition of the axiom $\forall x\, y\, .x\, R_{K_i}\, y \supset x\, R_O\, y$ to the theory of K
4. The addition of the following two rules of inference to LKM_K:

$$\frac{[\Re \Vdash^O w\, R_O\, x] \;\; \Re; (p)_x, \Sigma \longrightarrow \Theta}{\Re; (Op)_w, \Sigma \longrightarrow \Theta}\; \text{O-ia} \qquad \frac{\Re, w\, R_L\, x; \Sigma \longrightarrow (p)_x, \Theta}{\Re; \Sigma \longrightarrow (Op)_w, \Theta}\; \text{O-ia}^{\ddagger}$$

‡This rule contains the proviso that x does not appear as a variable on $\Re$ or an ornamentation of a formula in the lower sequent.

As an example of the flavor of this system, we present the proof of the three wise men as reported in [28]. The interested reader should refer to that paper and note the exact correspondence between that (semantic) argument and our proof.

Example 3.2 In the following, interpret p_i as the proposition, "wise man i has a white dot on his head." K_i is, of course, interpreted as, "wise man i knows that", and O is interpreted as, "it is common knowledge that." The axioms for the wise man puzzle (adapted from [28]) are:

1. $p_1 \wedge p_2 \wedge p_3$
2. $O(p_1 \vee p_2 \vee p_3)$
3. $O([K_1]p_3 \wedge [K_3]p_1 \wedge [K_1]p_2 \wedge [K_2]p_1 \wedge [K_2]p_3 \wedge [K_3]p_2)$[9]
4. $K_3(K_2(\sim K_1(p_1)))$
5. $K_3(\sim K_2(p_2))$

We will in general omit inference figures using thinning, since they are easily inserted from context. The proof is:

$$\dfrac{\text{closes}}{\Re_3; (p_1 \vee p_2 \vee p_3)_{w3} \longrightarrow (p_3)_{w3}(p_2)_{w3}, (p_1)_{w3}}\ \text{several } \vee\text{−IA}$$

$$\dfrac{\Re_3; (p_1 \vee p_2 \vee p_3)_{w3} \longrightarrow (p_3)_{w3}(p_2)_{w3}, (p_1)_{w3}}{\Re_3; 3_{w0} \longrightarrow (p_3)_{w3}(p_2)_{w3}, (p_1)_{w3}}\ \text{O-IA}$$

$$\dfrac{\Re_3; 3_{w0} \longrightarrow (p_3)_{w3}(p_2)_{w3}, (p_1)_{w3}}{\Re_3; 3_{w0}, K_1(\sim p_2)_{w2} \longrightarrow (p_2)_{w3}, (p_1)_{w3}}\ K_1\text{−IA}, \neg\text{−IA}$$

$$\dfrac{\Re_3; (p_3)_{w2} \longrightarrow (p_3)_{w2}}{\Re_3; K_1(p_3)_{w2} \longrightarrow (p_3)_{w2}}\ K_1\text{−IA}$$

$$\dfrac{\Re_3; K_1(p_3)_{w2} \longrightarrow (p_3)_{w2} \qquad \Re_3; 3_{w0}, K_1(\sim p_2)_{w2} \longrightarrow (p_2)_{w3}, (p_1)_{w3}}{\Re_3; 3_{w0}, (K_1(p_3) \vee K_1(\sim p_3))_{w2} \longrightarrow (p_2)_{w3}, (p_3)_{w2}, (p_1)_{w3}}\ \vee\text{−IA}$$

$$\dfrac{\Re_3; 3_{w0}, (K_1(p_3) \vee K_1(\sim p_3))_{w2} \longrightarrow (p_2)_{w3}, (p_3)_{w2}, (p_1)_{w3}}{\Re_3; 3_{w0}, (K_1(p_3) \vee K_1(\sim p_3))_{w2}, (K_1(\sim p_2))_{w2} \longrightarrow (p_3)_{w2}, (p_1)_{w3}}\ K_1\text{−IA}, \neg\text{−IA}$$

$$\dfrac{\Re_3; (p_2)_{w2} \longrightarrow (p_2)_{w2}}{\Re_3; K_1(p_2)_{w2} \longrightarrow (p_2)_{w2}}\ K_1\text{−IA}$$

$$\dfrac{\Re_3; K_1(p_2)_{w2} \longrightarrow (p_2)_{w2} \qquad \text{See above}}{\Re_3; 2_{w0}, (K_1(p_2) \vee K_1(\sim p_2))_{w2}, (K_1(p_3) \vee K_1(\sim p_3))_{w2} \longrightarrow (p_2)_{w2}, (p_3)_{w1}, (p_1)_{w3}}\ \vee\text{−IA}$$

$$\dfrac{\Re_3; 2_{w0}, (K_1(p_2) \vee K_1(\sim p_2))_{w2}, (K_1(p_3) \vee K_1(\sim p_3))_{w2} \longrightarrow (p_2)_{w2}, (p_3)_{w1}, (p_1)_{w3}}{\Re_3; 2_{w0}, 3_{w0} \longrightarrow (p_2)_{w2}, (p_3)_{w2}, (p_1)_{w3}}\ \text{o-ia}, \wedge\text{−IA}$$

$$\dfrac{\Re_3; 2_{w0}, 3_{w0} \longrightarrow (p_2)_{w2}, (p_3)_{w2}, (p_1)_{w3}}{\Re_3; 2_{w0}, 3_{w0} \longrightarrow (p_2)_{w2}, (p_3)_{w2}, (K_1(p_1))_{w2}}\ K_1\text{−IS} : \Re_3 := \Re_2 \cup \{w_2\, R_{K_1}\, w_3\}$$

$$\dfrac{\Re_3; 2_{w0}, 3_{w0} \longrightarrow (p_2)_{w2}, (p_3)_{w2}, (K_1(p_1))_{w2}}{\Re_2; 2_{w0}, 3_{w0} \sim K_1(p_1) \longrightarrow (p_2)_{w2}, (p_3)_{w2}}\ \neg\text{−IA}$$

$$\dfrac{\Re_2; 2_{w0}, 3_{w0} \sim K_1(p_1) \longrightarrow (p_2)_{w2}, (p_3)_{w2}}{\Re_2; 2_{w0}, 3_{w0}, 4_{w0} \longrightarrow (p_2)_{w2}, (p_3)_{w2}}\ K_2\text{−IA}, K_3\text{−IA}$$

$$\dfrac{\Re_2; 2_{w0}, 3_{w0}, 4_{w0} \longrightarrow (p_2)_{w2}, (p_3)_{w2}}{\Re_2; 2_{w0}, 3_{w0}, 4_{w0}, K_2(\sim p_3))_{w1} \longrightarrow (p_2)_{w2}}\ K_2\text{−IA}, \neg\text{−IA}$$

$$\dfrac{\Re_2; 2_{w0}, 3_{w0}, 4_{w0}, K_2(\sim p_3))_{w1} \longrightarrow (p_2)_{w2}}{\Re_2; 2_{w0}, 3_{w0}, 4_{w0}, K_2(\sim p_3))_{w1} \longrightarrow (K_2 p_2)_{w1}}\ K_2\text{−IS} : \Re_2 := \Re_1 \cup \{w_1\, R_{K_2}\, w_2\}$$

[9]Here we have adopted the convention that $[K_i]\alpha$ is shorthand for $K_i\alpha \vee K_i \sim \alpha$.

$$
\begin{array}{l}
\dfrac{\dfrac{\mathfrak{R}_1; (p_3)_{w1} \longrightarrow (p_3)_{w1}}{\mathfrak{R}_1; K_2(p_3)_{w1} \longrightarrow (p_3)_{w1}}\ K_2\text{–IA} \qquad \text{See above}}{\mathfrak{R}_1; 2_{w0}, 3_{w0}, 4_{w0}, (K_2(p_3) \vee K_2(\sim p_3))_{w1} \longrightarrow (K_2 p_2)_{w1}, (p_3)_{w1}}\ \vee\text{–IA} \\
\underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}\ \text{o-ia,several } \wedge\text{–IA} \\
\mathfrak{R}_1; 2_{w0}, 3_{w0}, 3_{w0}, 4_{w0} \longrightarrow (K_2 p_2)_{w1}, (p_3)_{w1} \\
\underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}\ \text{contraction} \\
\mathfrak{R}_1; 2_{w0}, 3_{w0}, 4_{w0} \longrightarrow (K_2 p_2)_{w1}, (p_3)_{w1} \\
\underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}\ K_3\text{–IA}, \neg\text{–IA} \\
\emptyset; 2_{w0}, 3_{w0}, 4_{w0}, 5_{w0} \longrightarrow (p_3)_{w1} \\
\underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}\ K_3\text{–IS} : \mathfrak{R}_1 := \emptyset \cup \{w0\, R_{K_3}\, w1\} \\
\emptyset; 2_{w0}, 3_{w0}, 4_{w0}, 5_{w0} \longrightarrow (K_3(p_3))_{w0} \\
\underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}\ \text{thinning} \\
\emptyset; 1_{w0}, 2_{w0}, 3_{w0}, 4_{w0}, 5_{w0} \longrightarrow (K_3(p_3))_{w0} \\
\underline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}\ \supset\text{–IS, several } \wedge\text{–IA} \\
\emptyset; \longrightarrow (1, 2, 3, 4, 5 \supset (K_3(p_3))_{w0}
\end{array}
$$

4 Linear Natural Deduction Proofs[10]

In this section we will show how to build natural deduction proofs in modal logic. These proofs will essentially be an incrementally constructed "linearization" of the previously presented Gentzen proofs. The justification of correctness will be based on sequent systems, and will in fact yield an algorithm for constructing Gentzen proofs corresponding the natural deduction proofs. As in the preceding section, we will first present a base collection of outline transforms paralleling the base collection of inference rules. We will establish correctness of this base set. We will then demonstrate the procedure for extending the system to the various modal calculi and demonstrate correctness for those languages. At the end of the section we present a (slightly edited) proof for the wisemen puzzle (see example 3.2).

4.1 The Base Collection of Proof Outline Transformations

A natural deduction proof will consist of a set of lines and a set of sequents, collectively referred to as a *proof outline*. The lines in the proof contain a line number, a set of hypotheses, a formula, and a justification. The line number is merely a label so that sets of hypotheses can be expressed as sets of numbers rather than lists of formulae. The combination of hypotheses, formula, and justification correspond to a sequent and the inference rule applied to derive this sequent from its predecessor. There is one special justification, NJ, which marks lines which for which a justification must be derived.

Definition 4.1 Let $\mathcal{L}$ be an ordered set of *line labels* and J be a set of *justifications* containing a special justification NJ. A *proof line* is an ordered tuple $\langle l, \mathcal{H}, \alpha, j \rangle$ where:

1. $l \in \mathcal{L}$,
2. $\mathcal{H} \subseteq \mathcal{L}$ where all members of $\mathcal{H}$ precede l,
3. α is some formula, and
4. $j \in J$ is the justification for this line.

[10]This section is primarily based on the notes presented in [20]

The sequents are used to keep track of the "leaves" of the (incomplete) Gentzen tree. As long as a branch is open, there will be a corresponding sequent in the proof outline. When a branch closes, the sequent will be deleted. The sequents are represented as sets of line labels rather than the actual formulae. Lines which appear on the left side of some sequent are called *supporting lines*, while lines appearing on the right side of a sequent are called *sponsoring lines*.

As long as there are sequents in the outline, there are sponsoring lines yet to be justified. Later in this section, we present outline transformations which will permit us to transform outlines into "more complete" outlines. This transforming process will finish when the outline is complete, *i.e.* when all branches close and the outline is actually a proof. Note that these proofs, like the Gentzen proofs they are derived from, will be *cut free* rather than axiomatic.

Definition 4.2 A *proof outline*, $\mathcal{O}$, is a pair, $\langle L, \Sigma \rangle$, where:

1. L is a list of proof lines which is a complete or incomplete ND-proof. A line with the justification NJ represents a piece of a proof which must be completed. Let L_0 be the set of all line labels in L which have this justification. These are called the *sponsoring* lines of $\mathcal{O}$.
2. $\Sigma = \{\Re; \Gamma_l \longrightarrow l \mid l \in L_0, \Gamma_l \subseteq L \setminus L_0\}$ is a set of sequents, where the line labels in Γ_l must precede l, and $\Re$ encodes the accessibility relation for this sequent. The lines in Γ_l, called *supporting* lines, and are said to *support* l; while, conversely, l *sponsors* the lines in Γ_l. A line is *active* if it is either a supporting line, or a sponsoring line which does not assert $\perp$.

It is easy to show that $\mathcal{O}$ has an active line if and only it Σ is not empty. We say that $\mathcal{O}$ is an outline *for A* if the last line in $\mathcal{O}$ (more precisely, in L) has no hypotheses and asserts $(A)_{w0}$. It is also easy to show that if line a supports line z then the hypotheses of a are a subset of the hypotheses of z.

A Gentzen proof is begun by writing the formula to be proved on the right side of the sequent arrow (with $\Re_0$ on the left side in our case) and applying operational inference rules until the tree closes or no rules are applicable. In an analogous fashion, we can define an initial proof outline, called the *trivial outline*, to which we apply *transformation rules* until the outline is complete.

Definition 4.3 Let A be a formula, and let z be the label for the proof line

$$(z) \quad \vdash \quad (A)_{w0} \qquad\qquad \text{NJ.}$$

If L is the list containing just z, and Σ is the set containing just the sequent $\Re_0; \longrightarrow z$ then $\mathcal{O}_0 := \langle L, \Sigma \rangle$ is clearly an outline. We call this outline the *trivial outline for A.*

Example 4.1 A proof outline for the theorem in Example 3.1 is gotten by setting $L = \langle 1,2,3,4,5,6 \rangle$, $\Sigma = \{w_1 R w_2, w_0 R w_1; 1, 2 \longrightarrow 3\}$ where the lines in L are:

(1)	1	$\vdash$	$(Lp)_{w0}$	Hyp
(2)	1	$\vdash$	$(p)_{w2}$	L-ded
(3)	1	$\vdash$	$(p)_{w2}$	NJ
(4)	1	$\vdash$	$(Lp)_{w1}$	L-gen
(5)	1	$\vdash$	$(LLp)_{w0}$	L-gen
(6)		$\vdash$	$(Lp \supset LLp)_{w0}$	deduct(5)

It is easy to verify that $\langle L, \Sigma \rangle$ is an outline. Also, by application of Rulep2 to lines 2 and 3, we get a completed outline. [11]

Below we list several transformations of outlines. These take an outline, $\mathcal{O} = \langle L, \Sigma \rangle$ in which Σ is not empty, and produce a new structure, $\mathcal{O}' = \langle L', \Sigma' \rangle$, which (as we shall verify) is also an outline. We shall assume that any sequent of the form $\Re; \Gamma \longrightarrow \bot$ is simply another way to write the sequent $\Re; \Gamma \longrightarrow$, *i.e* the sequent in which the succedent is empty.

In a Gentzen system, there are antecedent and succedent rules which introduce connectives into the formulae in the antecedent and succedent of the sequent respectively. Moreover, cut-free Gentzen systems have a *subformula property* which states that the formula occurring higher in the tree are simpler (contain fewer connectives) than the formulae lower in the tree. Hence, moving up the tree reduces the complexity of the formula appearing within sequents until axioms appear and the branch closes.

In outline transformations, there are *D-rules* and *P-rules* which perform a functions analogous to the antecedent and succedent rules of a Gentzen system respectively. The D- rules will be responsible for simplifying the logical complexity of support lines, while the P- rules simplify the logical complexity of sponsoring lines.

[11] A script of the session used to generate this proof appears in the appendix. The text processor output used to format the above example was also automatically generated.

A portion of the outline will be closed by the three transformations, RuleP, RuleP1, and RuleP2. These rules are responsible for giving a justification to a sponsoring line without creating a new sponsoring line. In this case, Σ' results from removing a sequent from Σ and justifying a sponsoring line.

The transformations below explicitly describe how to compute new members of Σ' from members of Σ. If a sequent, σ, in $\mathcal{O}$ is unaffected by the transformation, then we assume that $\sigma \in \Sigma'$. A similar description for computing L' from L is given by showing two boxes of proof lines separated by an arrow. The box on the left contains lines present in L, while the box on the right contains lines present in L'. If a line appears in the box on the right but not in the box on the left, we add this new line to L' in the position indicated by the alphabetical ordering of the line labels. If a line appears in both boxes, then its justification has been changed from NJ in L to a new justification in L'. It is always the case that all the lines in L are contained in L'.

If Σ' is not empty, then each sequent $\sigma' \in \Sigma'$ is of two kinds. If no line in σ' was altered or inserted by the transformation, then $\sigma' \in \Sigma$. Otherwise, σ' is *constructed* from a unique $\sigma \in \Sigma$. Some transformations, like D-Disj and P-Conj, will construct two sequents in Σ' from a sequent in Σ. Most of the D- and P- transformations will construct one sequent in Σ' from one in Σ.

Definition 4.4 (The Base Collection of Outlines)
D-Conj

Here a is a supporting line in $\mathcal{O}$. Σ' is the result of replacing a with the lines b, c everywhere in Σ, *i.e.* line a is no longer active.

(a)	$\mathcal{H}$	$\vdash$	$(A_1 \wedge A_2)_w$	RuleX

$\Longrightarrow$

(b)	$\mathcal{H}$	$\vdash$	$(A_1)_w$	RuleP: a
(c)	$\mathcal{H}$	$\vdash$	$(A_2)_w$	RuleP: a

D-Disj

Let a be a disjunctive support line and let line z be a sponsor for line a. Build Σ' by replacing the sequent $\Re; \Gamma_z \longrightarrow z$ with the two sequents $\Re; \Gamma, b \longrightarrow m$ and $\Re; \Gamma, n \longrightarrow y$, where $\Gamma := \Gamma_z \setminus \{a\}$.

(a)	$\mathcal{H}'$	$\vdash$	$(A_1 \lor A_2)_w$	RuleX
(z)	$\mathcal{H}$	$\vdash$	$(C)_{w'}$	NJ

$\Longrightarrow$

(b)	b	$\vdash$	$(A_1)_w$	Hyp
(m)	$\mathcal{H}, b$	$\vdash$	$(C)_{w'}$	NJ
(n)	n	$\vdash$	$(A_2)_w$	Hyp
(y)	$\mathcal{H}, n$	$\vdash$	$(C)_{w'}$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(C)_{w'}$	Cases: a, m, y

D-BackChain$_i$

Let $i = 1, 2$ and set $j := 3 - i$. Let a be a disjunctive support line which is sponsored by z, and let σ be the sequent $\Re; \Gamma, a \longrightarrow z$. If we let $\Sigma_0 := \Sigma \setminus \{\Re; \Gamma_z \longrightarrow z\}$ then $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma \longrightarrow m, \Re; \Gamma, n \longrightarrow x\}$.

(a)	$\mathcal{H}$	$\vdash$	$(A_1 \lor A_2)_w$	RuleX
(z)	$\mathcal{H}_1$	$\vdash$	$(C)_w$	NJ

$\Longrightarrow$

(m)	$\mathcal{H}_1$	$\vdash$	$(\neg A_i)_w$	NJ
(n)	n	$\vdash$	$(A_j)_w$	Hyp
(x)	$\mathcal{H}_1, n$	$\vdash$	$(C)_w$	NJ
(y)	$\mathcal{H}_1$	$\vdash$	$(A_j \supset C)_w$	Deduct: x
(z)	$\mathcal{H}_1$	$\vdash$	$(C)_w$	RuleP: a, m, y

D-ModusPonens

Let a be an implicational support line, which is sponsored by line z and let σ be the sequent $\Re; \Gamma, a \longrightarrow z$, Add the lines below to the outline. If we set $\Sigma_0 := \Sigma \setminus \{\Re; \Gamma_z \longrightarrow z\}$ then $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma \longrightarrow m,\ \Re; \Gamma, n \longrightarrow x\}$.

(a)	$\mathcal{H}$	$\vdash$	$(A_1 \supset A_2)_w$	RuleX
(z)	$\mathcal{H}_1$	$\vdash$	$(C)_w$	NJ

$\Longrightarrow$

(m)	$\mathcal{H}_1$	$\vdash$	$(A_1)_w$	NJ
(n)	n	$\vdash$	$(A_2)_w$	Hyp
(x)	$\mathcal{H}_1, n$	$\vdash$	$(C)_w$	NJ
(y)	$\mathcal{H}_1$	$\vdash$	$(A_2 \supset C)_w$	Deduct: x
(z)	$\mathcal{H}_1$	$\vdash$	$(C)_w$	RuleP: a, m, y

D-ModusTollens

The qualifications for this transformation are the same as those for D-ModusPonens.

(a)	$\mathcal{H}$	$\vdash$	$(A_1 \supset A_2)_w$	RuleX
(z)	$\mathcal{H}_1$	$\vdash$	$(C)_w$	NJ

$\Longrightarrow$

(m)	$\mathcal{H}_1$	$\vdash$	$(\neg A_2)_w$	NJ
(n)	n	$\vdash$	$(\neg A_1)_w$	Hyp
(x)	$\mathcal{H}_1, n$	$\vdash$	$(C)_w$	NJ
(y)	$\mathcal{H}_1$	$\vdash$	$(\neg A_1 \supset C)_w$	Deduct: x
(z)	$\mathcal{H}_1$	$\vdash$	$(C)_w$	RuleP: a, m, y

D-Imp

This rule treats implication as if it were an abbreviation of a disjunction. Σ' is the result of replacing a with b in each sequent of Σ. Line a is no longer active.

(a)	$\mathcal{H}$	$\vdash$	$(A_1 \supset A_2)_w$	RuleX

$\Longrightarrow$

(b)	$\mathcal{H}$	$\vdash$	$(\neg A_1 \lor A_2)_w$	RuleP

D-Neg
Apply one of the following four sub-transformations to line a, depending on which one matches the structure of a. Σ' is the result of replacing a with b in each sequent of Σ.

(a)	$\mathcal{H}$	$\vdash$	$(\neg\neg A)_w$	RuleX	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(A)_w$	RuleP: a
(a)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \vee A_2)_w$	RuleX	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(\neg A_1 \wedge \neg A_2)_w$	RuleP: a
(a)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \wedge A_2)_w$	RuleX	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(\neg A_1 \vee \neg A_2)_w$	RuleP: a
(a)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \supset A_2)_w$	RuleX	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(A_1 \wedge \neg A_2)_w$	RuleP: a

D-Thinning
If line a supports line z, then we can drop line a as a support of line z.

Each of the P-rules listed below will "process" a sponsoring line z. Let $\Sigma_0 := \Sigma \setminus \{\Re; \Gamma_z \longrightarrow z\}$.

P-Conj
Set $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma_z \longrightarrow m, \Re; \Gamma_z \longrightarrow y\}$.

						(m)	$\mathcal{H}$	$\vdash$	$(A_1)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(A_1 \wedge A_2)_w$	NJ	$\Longrightarrow$	(y)	$\mathcal{H}$	$\vdash$	$(A_2)_w$	NJ
						(z)	$\mathcal{H}$	$\vdash$	$(A_1 \wedge A_2)_w$	RuleP:m, y

P-Disj1
Set $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma_z, a \longrightarrow x\}$.

						(a)	a	$\vdash$	$(\neg A_1)_w$	Hyp
						(x)	$\mathcal{H}, a$	$\vdash$	$(A_2)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(A_1 \vee A_2)_w$	NJ	$\Longrightarrow$	(y)	$\mathcal{H}$	$\vdash$	$(\neg A_1 \supset A_2)_w$	Deduct: x
						(z)	$\mathcal{H}$	$\vdash$	$(A_1 \vee A_2)_w$	RuleP: y

P-Disj2
Set $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma_z, a \longrightarrow x\}$.

						(a)	a	$\vdash$	$(\neg A_2)_w$	Hyp
						(x)	$\mathcal{H}, a$	$\vdash$	$(A_1)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(A_1 \vee A_2)_w$	NJ	$\Longrightarrow$	(y)	$\mathcal{H}$	$\vdash$	$(\neg A_2 \supset A_1)_w$	Deduct: x
						(z)	$\mathcal{H}$	$\vdash$	$(A_1 \vee A_2)_w$	RuleP: y

P-Imp
Set $\Sigma' := \Sigma \cup \{\Re; \Gamma_z, a \longrightarrow y\}$.

						(a)	a	$\vdash$	$(A_1)_w$	Hyp
(z)	$\mathcal{H}$	$\vdash$	$(A_1 \supset A_2)_w$	NJ	$\Longrightarrow$	(y)	$\mathcal{H}, a$	$\vdash$	$(A_2)_w$	NJ
						(z)	$\mathcal{H}$	$\vdash$	$(A_1 \supset A_2)_w$	Deduct: y

P-Contrapositive

Set $\Sigma' := \Sigma \cup \{\Re; \Gamma_z, a \longrightarrow x\}$.

(z)	$\mathcal{H}$	$\vdash$	$(A_1 \supset A_2)_w$	NJ

$\Longrightarrow$

(a)	a	$\vdash$	$(\neg A_2)_w$	Hyp
(x)	$\mathcal{H}, a$	$\vdash$	$(\neg A_1)_w$	NJ
(y)	$\mathcal{H}$	$\vdash$	$(\neg A_2 \supset \neg A_1)_w$	Deduct: x
(z)	$\mathcal{H}$	$\vdash$	$(A_1 \supset A_2)_w$	RuleP: y

P-Neg

Apply one of the following four sub-transformations. Set $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma_z \longrightarrow y\}$.

(z)	$\mathcal{H}$	$\vdash$	$(\neg\neg A)_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(A)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\neg\neg A)_w$	RuleP: y

(z)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \vee A_2)_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(\neg A_1 \wedge \neg A_2)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \vee A_2)_w$	RuleP: y

(z)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \wedge A_2)_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(\neg A_1 \vee \neg A_2)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \wedge A_2)_w$	RuleP: y

(z)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \supset A_2)_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(A_1 \wedge \neg A_2)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\neg.A_1 \supset A_2)_w$	RuleP: y

P-Indirect

Set $\Sigma' := \Sigma \cup \{\Re; \Gamma_z, a \longrightarrow y\}$.

(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	NJ

$\Longrightarrow$

(a)	a	$\vdash$	$(\neg A)_w$	Hyp
(y)	$\mathcal{H}, a$	$\vdash$	$\bot$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	IP: y

P-Thinning

We may replace line z with line y as a sponsoring line, provided that what we get is still an outline. In this case, set $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma_z \longrightarrow y\}$, where $\Sigma_0 := \Sigma \setminus \{\Re; \Gamma_z \longrightarrow z\}$.

(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$\bot$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	RuleP: y

The following three versions of the RuleP transformation are used to complete a subproof, *i.e.* they remove a sequent from the list of the outline's sequents. RuleP1 and RuleP2 are included here for technical reasons to be made clear later. They are obviously subsumed by RuleP.

RuleP

Let lines $a_1, \ldots, a_n$ be some of the supports of z, such that $[\Re_z \wedge (A_1)_{w1} \wedge \ldots \wedge (A_n)_{wn}] \supset (A)_w$ is tautologous. Then we can change the justification of line z from NJ to RuleP: $a_1, \ldots, a_n$. Σ' is Σ_0.

(a_1)	$\mathcal{H}_1$	$\vdash$	$(A_1)_{w_1}$	RuleX
	$\vdots$	$\vdots$		$\vdots$
(a_n)	$\mathcal{H}_n$	$\vdash$	$(A_n)_{w_n}$	RuleX
(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	NJ

$\Longrightarrow$

(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	RuleP: $a_1, \ldots, a_n$

RuleP1

If A_1 and A_2 are complementary, signed atoms, then we have proved line z indirectly. Set $\Sigma' := \Sigma_0$.

(a_1)	$\mathcal{H}_1$	$\vdash$	$(A_1)_w$	RuleX
(a_2)	$\mathcal{H}_2$	$\vdash$	$(A_2)_w$	RuleY
(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	NJ

$\Longrightarrow$

(z)	$\mathcal{H}$	$\vdash$	$(A)_w$	RuleP: a_1, a_2

RuleP2

If A_1 and A_2 are signed atoms which are equal to each other, then we have proved line z directly. Set $\Sigma' := \Sigma_0$.

(a)	$\mathcal{H}_1$	$\vdash$	$(A_1)_w$	RuleX
(z)	$\mathcal{H}_2$	$\vdash$	$(A_2)_w$	NJ

$\Longrightarrow$

(z)	$\mathcal{H}_2$	$\vdash$	$(A_2)_w$	RuleP: a

Notice that after a D- transformation is applied, the line a may or may not still be active, while after a P- transformation is applied, the line z is no longer active.

4.2 Correctness of the Base Collection

In order to show that natural deduction proofs built using the above defined outline transformations are correct, we will show that:

1. The rules $\neg$—EA and $\neg$—ES which eliminate negation for the antecedent and succedent are admissible to LKM, and in fact give a procedure for removing those rules from a completed proof.

2. For each application of an outline transformation, there is a corresponding piece of an LKM proof which is a derivation of the new sequent in the outline from the old sequent.

3. Each use of the various forms of rulep remove a sequent only if there is an LKM proof for that branch.

There is a slight technical problem in that the correspondence between outline transformations and Gentzen proofs is not complete. In particular, the D-Neg rules manipulate formulae in a way which is not directly justifiable in terms of LKM inferences. To remedy this, we introduce the following three outline transformation rules. These are essentially composite transformations based on those given in Section 4.

Definition 4.5 Let D-Imp* be the transformation which results from combining D-Imp and D-Disj, *i.e.* apply D-Disj to the disjunctive line produced by D-Imp. Let D-Neg* be the transformation which does the following: applies D-Neg, and if that instance of the transformation was not used to remove double negation, then applies either D-Conj, D-Disj, or D-Imp, depending on the structure of the proof line resulting from the D-Neg application. Let P-Neg* be the transformation which does the following: applies P-Neg, and if that instance of the transformation was not used to remove double negation, then applies either P-Conj, P-Disj, or P-Imp depending on the structure of the proof line resulting from the P-Neg application.

Definition 4.6 Consider the following class of outline transformations: D-Conj, D-Disj, D-Imp*, D-Neg*, P-Conj, P-Disj1, P-Imp, P-Neg*, RuleP1, RuleP2, and P-Indirect. We shall call this collection the *minimal collection* of transformations.

Notice that Σ' contains fewer sequents than Σ if and only if the transformation rule applied was RuleP1 or RuleP2, and therefore, $\Sigma' \subset \Sigma$. This is simple to verify by checking that all the D- and P- transformations in the minimal collection never decrease the number of elements in the sequent set. RuleP1 and RuleP2, however, do decrease this number by one, by removing a sequent (*i.e.* supplying a justification to an NJ line).

In the following, let LKM* denote the base collection plus the rules $\neg$−EA and $\neg$−ES.

Proposition 4.1 If some sequent has a proof in LKM*, it has a proof in LKM.

> **Proof:** The proof is a simple induction-like procedure showing the elimination rules can be moved up the tree and finally eliminated.
>
> - For the ground case, consider a tree of height one, where the elimination rules lead to an axiom. Then we have:
>
> $$\frac{\neg p \longrightarrow \neg p}{\neg p, p \longrightarrow} \ \neg\text{−ES}$$

converts to

$$\frac{p \longrightarrow p}{\neg p, p \longrightarrow} \; \neg\text{–IA}$$

and similarly for ¬–EA.

- Note the elimination rules can be "slid" past all introduction rules, other than ¬–IA in the case of ¬–ES and ¬–IS in the case of ¬–EA, by virtue of the fact that those rules cannot interfere with the applicability of the elimination rules. We will demonstrate this for ∧–IA.

$$\dfrac{\dfrac{\Re; \Gamma, (p)_w, (r)_{w'} \longrightarrow \Theta}{\Re; \Gamma, (p \wedge q)_w (r)_{w'} \longrightarrow \Theta} \; \wedge\text{–IA}}{\Re; \Gamma, (p \wedge q)_w, (\neg r)_{w'} \longrightarrow \Theta} \; \neg\text{–ES}$$

converts to

$$\dfrac{\dfrac{\Re; \Gamma, (p)_w, (r)_{w'} \longrightarrow \Theta}{\Re; \Gamma, (p)_w, (\neg r)_{w'} \longrightarrow \Theta} \; \neg\text{–ES}}{\Re; \Gamma, (p \wedge q)_w, (\neg r)_{w'} \longrightarrow \Theta} \; \wedge\text{–IA}$$

- For the negation introduction rules, we have the following cases:

$$\dfrac{\dfrac{\dfrac{\cdots\cdot}{\Gamma, p \longrightarrow \Delta} \; \text{RuleX}}{\Gamma \longrightarrow \neg p, \Delta} \; \neg\text{–IA}}{\Gamma, p \longrightarrow \Delta} \; \neg\text{–ES}$$

converts to

$$\dfrac{\cdots\cdot}{\Gamma, p \longrightarrow \Delta} \; \text{RuleX}$$

and similarly for ¬–EA.

♣

Proposition 4.2 Assume that $\mathcal{O}' = \langle L', \Sigma' \rangle$ is the result of applying some transformation T_i from the minimal collection of transformations to the outline $\mathcal{O} = \langle L, \Sigma \rangle$. If Σ' is properly contained in Σ and $\Re; \Gamma_z \longrightarrow z$ is a member of Σ and not of Σ', then $\Re; \Gamma_z \longrightarrow z$ has a cut-free, LKM*-proof.

Proof: Assume the hypotheses of this theorem. Clearly the transformation applied to $\mathcal{O}$ was either RuleP1 or RuleP2.

Assume that it was RuleP1. Then Γ_z must have contained two lines, say a and b which asserted $(A)_w$ and $(\neg A)_w$, respectively, where $(A)_w$ is an atomic formula. Let $(C)_{w'}$ be the assertion in line z, and let $\Gamma := \Gamma_z \setminus \{a, b\}$. Then a cut-free, LKM*-proof of $\Re; \Gamma_z \longrightarrow z$ is the following.

$$\dfrac{\dfrac{\Re; (A)_w \longrightarrow (A)_w}{\Re; (A)_w, (\neg A)_w \longrightarrow} \neg\text{-IA}}{\Re; \Gamma, (A)_w, (\neg A)_w \longrightarrow (C)_{w'}} \text{ several Thinning}$$

Assume that it was RuleP2. Then Γ_z must have contained a line say a which asserts $(A)_w$, where $(A)_w$ is an atomic formula and such that z asserts $(A)_w$. Then a cut-free, LKM*-proof of $\Re; \Gamma_z \longrightarrow z$ is simply the following. Here, $\Gamma := \Gamma_z \setminus \{a\}$.

$$\dfrac{\Re; (A)_w \longrightarrow (A)_w}{\Re; \Gamma, (A)_w \longrightarrow (A)_w} \text{ several Thinning}$$

♣

Proposition 4.3 Let $\mathcal{O}'$ be the result of applying one of the transformations in the minimal collection to the outline $\mathcal{O}$. Let Σ and Σ' be the sequent sets associated with $\mathcal{O}$ and $\mathcal{O}'$. If each sequent in Σ' has a cut-free, LKM*-proof then each sequent in Σ has a cut-free, LKM*-proof. We will refer to an outline transformation having this property as *correct*.

Proof: If the transformation which was applied was either RuleP1 or RuleP2, then the preceding Proposition says that the sequent removed from Σ has an cut-free, LKM*-proof. In the cases where a D- or P-transformation was applied, either one or two sequents in Σ' are constructed from a sequent in Σ. (More than one or two sequents in Σ' may have been constructed, however, from the application of some transformations.) Below we show how to combine cut-free, LKM*-proofs for those one or two sequents to give a cut-free, LKM*-proof of the original sequent in Σ. Let $(C)_{w'}$ denote the formula asserted by a line supported by $(A)_w$. We shall not specify when the inference rule of interchange is used, since it will be easy for the reader to insert them in the inference figure where they are required.

Case D-Conj:

$$\cfrac{\cfrac{\cfrac{\Re;\Gamma,(A_1)_w,(A_2)_w \longrightarrow (C)_{w'}}{\Re;\Gamma,(A_1\wedge A_2)_w,(A_2)_w \longrightarrow (C)_{w'}}\;\wedge\text{-IA}}{\Re;\Gamma,(A_1\wedge A_2)_w,(A_1\wedge A_2)_w \longrightarrow (C)_{w'}}\;\wedge\text{-IA}}{\Re;\Gamma,(A_1\wedge A_2)_w \longrightarrow (C)_{w'}}\;\text{Contraction}$$

Case D-Disj:

$$\cfrac{\Re;\Gamma,(A_1)_w \longrightarrow (C)_{w'} \qquad \Re;\Gamma,(A_2)_w \longrightarrow (C)_{w'}}{\Re;\Gamma,(A_1\vee A_2)_w \longrightarrow (C)_{w'}}\;\vee\text{-IA}$$

Case D-Imp*:

$$\cfrac{\cfrac{\Re;\Gamma,(\neg A_1)_w \longrightarrow (C)_{w'}}{\Re;\Gamma \longrightarrow (C)_{w'},(A_1)_w}\;\neg\text{-EA} \qquad \Re;\Gamma,(A_2)_w \longrightarrow (C)_{w'}}{\Re;(A_1\supset A_2)_w,\Gamma \longrightarrow (C)_{w'}}\;\supset\text{-IA}$$

Case D-Neg: If a asserts $(\neg\neg A)_w$ then

$$\cfrac{\cfrac{\Re;\Gamma,(A)_w \longrightarrow (C)_{w'}}{\Re;\Gamma \longrightarrow (C)_{w'},(\neg A)_w}\;\neg\text{-IS}}{\Re;\Gamma,(\neg\neg A)_w \longrightarrow (C)_{w'}}\;\neg\text{-IA}$$

If a asserts $(\neg.A_1\vee A_2)_w$ then

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\Re;\Gamma,(\neg A_1)_w,(\neg A_2)_w \longrightarrow (C)_{w'}}{\Re;\Gamma,(\neg A_2)_w \longrightarrow (C)_{w'},(A_1)_w}\;\neg\text{-EA}}{\Re;\Gamma \longrightarrow (C)_{w'},(A_1)_w,(A_2)_w}\;\neg\text{-EA}}{\Re;\Gamma \longrightarrow (C)_{w'},(A_1\vee A_2)_w,(A_2)_w}\;\vee\text{-IS}}{\Re;\Gamma \longrightarrow (C)_{w'},(A_1\vee A_2)_w,(A_1\vee A_2)_w}\;\vee\text{-IS}}{\Re;\Gamma \longrightarrow (C)_{w'},(A_1\vee A_2)_w}\;\text{Contraction}}{\Re;\Gamma,(\neg.A_1\vee A_2)_w \longrightarrow (C)_{w'}}\;\neg\text{-IA}$$

If a asserts $(\neg.A_1\wedge A_2)_w$ then

$$\dfrac{\dfrac{\Re;\Gamma,(\neg A_1)_w \longrightarrow (C)_{w'}}{\Re;\Gamma \longrightarrow C_{w'},(A_1)_w}\;\neg\text{–EA} \qquad \dfrac{\Re;\Gamma,(\neg A_2)_w \longrightarrow (C)_{w'}}{\Re;\Gamma \longrightarrow C_{w'},(A_2)_w}\;\neg\text{–EA}}{\dfrac{\Re;\Gamma \longrightarrow C_{w'},(A_1 \wedge A_2)_w}{\Re;\Gamma,(\neg.A_1 \wedge A_2)_w \longrightarrow (C)_{w'}}\;\neg\text{–IA}}\;\wedge\text{–IA}$$

If a asserts $(\neg.A_1 \supset A_2)_w$ then

$$\dfrac{\dfrac{\dfrac{\Re;\Gamma,(A_1)_w,(\neg A_2)_w \longrightarrow (C)_{w'}}{\Re;\Gamma,(A_1)_w \longrightarrow C_{w'},(A_2)_w}\;\neg\text{–EA}}{\Re;\Gamma \longrightarrow C_{w'},(A_1 \supset A_2)_w}\;\supset\text{–IS}}{\Re;\Gamma,(\neg.A_1 \supset A_2)_w \longrightarrow (C)_{w'}}\;\neg\text{–IA}$$

Case P-Conj:

$$\dfrac{\Re;\Gamma \longrightarrow (A_1)_w \qquad \Re;\Gamma \longrightarrow (A_2)_w}{\Re;\Gamma \longrightarrow (A_1 \wedge A_2)_w}\;\wedge\text{–IA}$$

Case P-Disj1:

$$\dfrac{\dfrac{\dfrac{\Re;\Gamma,(\neg A_1)_w \longrightarrow (A_2)_w}{\Re;\Gamma \longrightarrow (A_1)_w,(A_2)_w}\;\neg\text{–EA}}{\Re;\Gamma \longrightarrow (A_1 \vee A_2)_w,(A_1 \vee A_2)_w}\;\vee\text{–IS, twice}}{\Re;\Gamma \longrightarrow (A_1 \vee A_2)_w}\;\text{Contraction}$$

Case P-Imp:

$$\dfrac{\Re;\Gamma,(A_1)_w \longrightarrow (A_2)_w}{\Re;\Gamma \longrightarrow (A_1 \supset A_2)_w}\;\supset\text{–IS}$$

Case P-Neg:

If z asserts $(\neg\neg A)_w$, then

$$\dfrac{\dfrac{\Re;\Gamma \longrightarrow (A)_w}{\Re;\Gamma,(\neg A)_w \longrightarrow}\;\neg\text{–IA}}{\Re;\Gamma \longrightarrow (\neg\neg A)_w}\;\neg\text{–IS}$$

If z asserts $(\neg.A_1 \vee A_2)_w$ then

$$\cfrac{\cfrac{\mathfrak{R};\Gamma \longrightarrow (\neg A_1)_w}{\mathfrak{R};\Gamma,(A_1)_w \longrightarrow}\ \neg\text{-ES} \qquad \cfrac{\mathfrak{R};\Gamma \longrightarrow (\neg A_2)_w}{\mathfrak{R};\Gamma,(A_2)_w \longrightarrow}\ \neg\text{-ES}}{\cfrac{\mathfrak{R};\Gamma,(A_1 \vee A_2)_w \longrightarrow}{\mathfrak{R};\Gamma \longrightarrow (\neg.A_1 \vee A_2)_w}\ \neg\text{-IS}}\ \vee\text{-IA}$$

If z asserts $(\neg.A_1 \wedge A_2)_w$ then

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\mathfrak{R};\Gamma,(\neg\neg A_1)_w \longrightarrow (\neg A_2)_w}{\mathfrak{R};\Gamma \longrightarrow (\neg A_2)_w,(\neg A_1)_w}\ \neg\text{-EA}}{\mathfrak{R};\Gamma,(A_1)_w \longrightarrow (\neg A_2)_w}\ \neg\text{-ES}}{\mathfrak{R};\Gamma,(A_1)_w,(A_2)_w \longrightarrow}\ \neg\text{-ES}}{\mathfrak{R};\Gamma,(A_1 \wedge A_2)_w,(A_1 \wedge A_2)_w \longrightarrow}\ \wedge\text{-IA twice}}{\mathfrak{R};\Gamma,(A_1 \wedge A_2)_w \longrightarrow}\ \text{Contraction}}{\mathfrak{R};\Gamma \longrightarrow (\neg.A_1 \wedge A_2)_w}\ \neg\text{-IS}$$

If z asserts $(\neg.A_1 \supset A_2)_w$ then

$$\cfrac{\cfrac{\cfrac{\mathfrak{R};\Gamma \longrightarrow (A_1)_w \qquad \cfrac{\mathfrak{R};\Gamma \longrightarrow (\neg A_2)_w}{\mathfrak{R};\Gamma,(A_2)_w \longrightarrow}\ \neg\text{-ES}}{\mathfrak{R};\Gamma,\Gamma,(A_1 \supset A_2)_w \longrightarrow}\ \supset\text{-IA}}{\mathfrak{R};\Gamma,(A_1 \supset A_2)_w \longrightarrow}\ \text{several Contractions}}{\mathfrak{R};\Gamma \longrightarrow (\neg.A_1 \supset A_2)_w}\ \neg\text{-IA}$$

Case P-Indirect:

$$\cfrac{\mathfrak{R};\Gamma,(\neg A)_w \longrightarrow}{\mathfrak{R};\Gamma \longrightarrow (A)_w}\ \neg\text{-EA}$$

♣

Proposition 4.4 Let A be a formula, and let $\mathcal{O}_0$ be the trivial outline for A. If there is some list $\langle T_1, \ldots, T_n \rangle$ of transformations from the minimal collection of transformation such that $\mathcal{O} := T_n(\ldots(T_1(\mathcal{O}_0)\ldots)$ contains an empty sequent set, then the lines in $\mathcal{O}$ form a completed ND-proof of A and A has an LKM-proof. Furthermore, this LKM-proof can easily be constructed by using the constructions given in the proofs of Propositions 4.1, 4.2, and 4.3.

Proof: Is an immediate consequence of the preceding propositions. ♣

Extensions to this basic system will involve adding more outline transforms to the base collection. Soundness of these systems will be based solely on the correctness of the new outline transforms with implicit reference to the preceding proposition.

4.3 Outline Transformations for Logics of Knowledge

We will now extend the base set of transformations to encompass modal operators. Since the classical systems are contained in the knowledge systems, we will give rules for the knowledge systems and establish correctness for those rules.

As it turns out, there are two different sets of outline transformations possible for modal operators in $\mathcal{R}$ depending on whether the dual modality is present in Ω. We will consider both those cases below. The justification for the modalities is based on whether the introduction of the modal operator forces the generation of a new related world, or uses the structure contained on $\mathfrak{R}$ to deduce a relationship.

Definition 4.7 Let the set of outline transformation rules for the modal operators *with* dual modalities be comprised of the base collection plus the following rules for each $\rho \in \mathcal{R}$ and $\omega \in \Omega$.

D-ρ
Let a be a modal support line, and suppose that $w\, R_\rho\, w'$ follows from $\mathfrak{R}$. Σ' is Σ with a replaced by a, b, i.e line a stays active.

(a)	$\mathcal{H}$	$\vdash$	$(\rho(p))_w$	RuleX	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(p)_{w'}$	ρ-ded: a

D-ω
Let a be a modal support line, and w' is a new world not appearing in $\mathfrak{R}$. Σ' is the result of replacing a by b, and adding $w\, R_\rho\, w'$ to $\mathfrak{R}$ in Σ.

						(b)	b	$\vdash$	$(p)_{w'}$	Hyp
(a)	$\mathcal{H}$	$\vdash$	$(\omega(p))_w$	RuleX	$\Longrightarrow$	(y)	$\mathcal{H}, b$	$\vdash$	$(C)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(C)_w$	NJ		(z)	$\mathcal{H}$	$\vdash$	$(C)_w$	ω-gen: a, y

D-Neg

We also need the following two transforms which are cases of D-Neg in the base collection:

(a)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_w$	RuleX

$\Longrightarrow$

(b)	$\mathcal{H}$	$\vdash$	$(\omega\neg p)_w$	RuleQ: a

(a)	$\mathcal{H}$	$\vdash$	$(\neg\omega(p))_w$	RuleX

$\Longrightarrow$

(b)	$\mathcal{H}$	$\vdash$	$(\rho\neg p)_w$	RuleQ: a

P-ρ

Let z be a modal sponsoring line and let w' be some world which is not free in $\Re$. Set $\Sigma' := \Sigma_0 \cup \{\Re, w\, R_\rho\, w'; \Gamma_z \longrightarrow y\}$.

(z)	$\mathcal{H}$	$\vdash$	$(\rho(p))_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(p)_{w'}$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\rho(p))_w$	ρ-gen:y

P-ω1

In this case, z is a modal sponsoring line and $w\, R_\rho\, w'$ follows from $\Re$.
Let $\Sigma' := \Sigma_0 \cup \{\ \Re; \Gamma_z \ \longrightarrow\ y\}$.

(z)	$\mathcal{H}$	$\vdash$	$(\omega(p))_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(p)_{w'}$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\omega(p))_w$	ω-ded: y

P-ω2

In this case, z is a modal sponsoring line and $w\, R_\rho\, w'$ follows from $\Re$. Here, we allow the formula of the sponsoring line to be used again by inserting it the outline as a hypothesis and processing the new sponsoring line by contradiction. Let $\Sigma' := \Sigma_0 \cup \{\Re; \Gamma_z, a, b \ \longrightarrow\ y\}$.

(z)	$\mathcal{H}$	$\vdash$	$(\omega(p))_w$	NJ

$\Longrightarrow$

(a)	a	$\vdash$	$(\neg\omega(p))_w$	Hyp
(b)	$\mathcal{H}$	$\vdash$	$\neg(p)_{w'}$	ω-ded: a
(y)	$\mathcal{H}, a$	$\vdash$	$\bot$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\omega(p))_w$	IP:y

P-Neg

We also need the following two transforms under the same conditions as P-Neg of the base collection:

(z)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(\omega\neg p)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_w$	RuleQ: y

(z)	$\mathcal{H}$	$\vdash$	$(\neg\omega(p))_w$	NJ

$\Longrightarrow$

(y)	$\mathcal{H}$	$\vdash$	$(\rho\neg p)_w$	NJ
(z)	$\mathcal{H}$	$\vdash$	$(\neg\omega(p))_w$	RuleQ: y

If there are modal operators for which the dual does not exist in the language, then we have to treat negative modalities containing that modal operator in a fashion similar to that found in the D-Neg* rules of the correctness proof. In other words, instead of using a negation rule to move the negation inward and then applying the appropriate rule for the modality, we will do both at once.

Definition 4.8 Let the set of outline transformation rules for the modal operators *without* dual modalities be comprised of the base collection plus the following rules for each $\rho \in \mathcal{R}$[12]

D-ρ

Let a be a modal support line, and suppose that $w\, R_\rho\, w'$ follows from $\mathfrak{R}$. Σ' is Σ with a replaced by a, b wherever it occurs, i.e. line a stays active.

(a)	$\mathcal{H}$	$\vdash$	$(\rho(p))_w$	RuleX	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(p)_{w'}$	ρ-ded: a

D-Neg

We need the following case of the D-Neg rules given in the base collection. Here w' is some new world. Replace a by b wherever it occurs in Σ to get Σ'.

(a)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_w$	RuleX	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(\neg p)'_w$	ρ-gen: a

P-ρ

Let z be a modal sponsoring line and let w' be some world which is not free in $\mathfrak{R}$. Set $\Sigma' := \Sigma_0 \cup \{\mathfrak{R}, w\, R_\rho\, w'; \Gamma_z \longrightarrow y\}$.

(z)	$\mathcal{H}$	$\vdash$	$(\rho(p))_w$	NJ	$\Longrightarrow$	(y)	$\mathcal{H}$	$\vdash$	$(p)_{w'}$	NJ
						(z)	$\mathcal{H}$	$\vdash$	$(\rho(p))_w$	ρ-gen:y

P-Neg1

We also need following case of P-Neg where $\mathfrak{R} \vdash wRw'$, and $\Sigma' := \Sigma_0 \cup \{\ \mathfrak{R}; \Gamma_z \ \longrightarrow\ y\}$

(z)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_{w'}$	NJ	$\Longrightarrow$	(y)	$\mathcal{H}$	$\vdash$	$(\neg p)_w$	NJ
						(z)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_w$	ρ-ded:y

P-Neg2

Finally, we need following case of P-Neg where $\mathfrak{R} \vdash wRw'$: In this case $\Sigma' := \Sigma_0 \cup \{\mathfrak{R}; \Gamma_z, a, b \ \longrightarrow\ y\}$

						(a)	a	$\vdash$	$(\rho(p))_w$	Hyp
(z)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_{w'}$	NJ	$\Longrightarrow$	(b)	$\mathcal{H}$	$\vdash$	$(p)_{w'}$	ρ-ded: a
						(y)	$\mathcal{H}, b$	$\vdash$	$\bot$	NJ
						(z)	$\mathcal{H}$	$\vdash$	$(\neg\rho(p))_w$	IP:y

Proposition 4.5 The rules given in definitions 4.7 and 4.8 are correct.

[12]In order not to belabor the presentation, we consider only universal modalities. We leave it to the reader to supply the outline transformations for existential modalities having no dual in the language.

Proof:

Since the rules of definition 4.8 are the D-Neg* rules for definition 4.7, the following proof establishes the claim.

Case D-ρ:

$$\cfrac{\cfrac{\Re;\Gamma,(\rho p)_w,(p)_{w'} \longrightarrow (C)_{w1}}{\Re;\Gamma,(\rho p)_w,(\rho p)_w \longrightarrow (C)_{w1}}\ \rho\text{—IA}}{\Re;\Gamma,(\rho p)_w \longrightarrow (C)_{w1}}\ \text{Contraction}$$

Case D-ω:

$$\cfrac{\Re, w\,R_\rho\,w';\Gamma,(p)_{w'} \longrightarrow (C)_{w1}}{\Re;\Gamma,(\omega p)_w \longrightarrow (C)_{w1}}\ \omega\text{—IA}$$

The proviso on w' set by ω—IA is satisfied by the condition of D-ω, and the proviso on $w\,R_\rho\,w'$ set by ρ—IA is satisfied by the condition on D-ρ.

Case D-Neg of definition 4.8 and D-Neg* for definition 4.7:
If a asserts $(\neg\rho p)_w$ then

$$\cfrac{\cfrac{\cfrac{\Re, w\,R_\rho\,w';\Gamma,(\neg p)_{w'} \longrightarrow (C)_{w1}}{\Re, w\,R_\rho\,w';\Gamma \longrightarrow C_{w1},(p)_{w'}}\ \neg\text{—EA}}{\Re;\Gamma \longrightarrow C_{w'},(\rho p)_w}\ \rho\text{—IS}}{\Re;\Gamma,(\neg\rho p)_w \longrightarrow (C)_{w1}}\ \neg\text{—IA}$$

If a asserts $(\neg\omega p)_w$ then

$$\cfrac{\cfrac{\cfrac{\cfrac{\Re;\Gamma,(\neg\omega p)_w,(\neg p)_{w'} \longrightarrow (C)_{w1}}{\Re;\Gamma,(\neg\omega p)_w \longrightarrow (C)_{w1},p}\ \neg\text{—EA}}{\Re;\Gamma,(\neg\omega p)_w \longrightarrow (C)_{w1},Mp}\ \omega\text{—IS}}{\Re;\Gamma,(\neg\omega p)_w,(\neg\omega p)_w \longrightarrow (C)_{w1}}\ \neg\text{—IA}}{\Re;\Gamma,(\neg\omega p)_w \longrightarrow (C)_{w1}}\ \text{Contraction}$$

As in the D-ω and D-ρ cases, the provisos on ρ—IS and ω—IS are met by the conditions on D-ρ and D-ω.

Case P-ρ:

$$\cfrac{\Re, w\,R_\rho\,w';\Gamma \longrightarrow (p)_{w'}}{\Re;\Gamma \longrightarrow (\rho p)_w}\ \rho\text{—IS}$$

Case P-ω:

$$\frac{\dfrac{\Re;\Gamma \longrightarrow (p)'_w}{\Re;\Gamma \longrightarrow (\omega p)_w, (\omega p)_w}\ \omega\text{—IS}}{\Re;\Gamma \longrightarrow (\omega p)_w}\ \text{contraction}$$

As in the D-ω and D-ρ cases, the provisos on the rules are satisfied by the restrictions on the outline transformation.

Case P-Neg of definition 4.8 and P-Neg* of definition 4.7:

If z asserts $(\neg\omega p)_w$ then

$$\frac{\dfrac{\dfrac{\Re, w\, R_\rho\, w';\Gamma \longrightarrow (\neg p)_{w'}}{\Re, w\, R_\rho\, w';\Gamma, (p)_{w'} \longrightarrow}\ \neg\text{—ES}}{\Re;\Gamma, (\omega p)_w \longrightarrow}\ \omega\text{—IA}}{\Re;\Gamma \longrightarrow (\neg\omega p)_w}\ \neg\text{—IS}$$

If z asserts $(\neg\rho p)_w$, then (for the general case)

$$\frac{\dfrac{\dfrac{\Re, w\, R_\rho\, w';\Gamma, (\rho p)_w, (p)_{w'} \longrightarrow}{\Re;\Gamma, (\rho p)_w, (\rho p)_w \longrightarrow}\ \rho\text{—IA}}{\Re;\Gamma, (\rho p)_w \longrightarrow}\ \text{contraction}}{\Re;\Gamma \longrightarrow \neg(\rho p)_w}\ \neg\text{—ES}$$

The provisos on ρ—IA and ω—IA are (again) satisfied by the conditions on the corresponding outline transformations.

♣

Example 4.2 The following is a partially completed (and somewhat edited) version of the wise man problem presented in example 3.2[13]. The reader should refer to that example for the interpretation of the propositions and modal operators, and the axiomitization of the puzzle. Also, note that a set of outline transformations for the O modality were included with provisos corresponding to those stated in Definition 3.4.

First we have broken down all of the initial information by applications of propositional rules, and employed the common knowledge transformations to deduce lines 7 and 8.

[13]The initial version of this proof was automatically generated, and that version edited for readability.

(1)	1	$\vdash$	$(1 \wedge 2 \wedge 3 \wedge 4 \wedge 5)_{w0}$	Hyp
(3)	1	$\vdash$	$(2 \wedge 3 \wedge 4 \wedge 5)_{w0}$	Rulep(1)
(4)	1	$\vdash$	$(O(p_1 \vee p_2 \wedge p_3))_{w0}$	Rulep(3)
(6)	1	$\vdash$	$(O([K_1]p_3) \wedge [K_3]p_1 \wedge ([K_1]p_2) \wedge [K_2]p_1 \wedge [K_2]p_3 \wedge [K_3]p_2)_{w0}$	Rulep(5)
(7)	1	$\vdash$	$([K_1]p_3) \wedge [K_3]p_1 \wedge ([K_1]p_2) \wedge [K_2]p_1 \wedge [K_2]p_3 \wedge [K_3]p_2)_{w1}$	O-ia(6)
(8)	1	$\vdash$	$([K_1]p_3) \wedge [K_3]p_1 \wedge ([K_1]p_2) \wedge [K_2]p_1 \wedge [K_2]p_3 \wedge [K_3]p_2)_{w2}$	O-ia(6)
(11)	1	$\vdash$	$(K_1p_3 \vee K_1\neg p_3)_{w2}$	Rulep(8)
(12)	1	$\vdash$	$(K_1p_2 \vee K_1\neg p_2)_{w2}$	Rulep(8)
(16)	1	$\vdash$	$(K_2p_3 \vee K_2\neg p_3)_{w1}$	Rulep(15)
(18)	1	$\vdash$	$(K_3K_2\neg K_1p_1 \wedge K_3\neg K_2p_2)_{w0}$	Rulep(1)

The lines below are using modality rules to infer new information about what propositions are known by whom.

(19)	1	$\vdash$	$(K_3K_2\neg K_1p_1)_{w0}$	Rulep(18)
(20)	1	$\vdash$	$(K_2\neg K_1p_1)_{w1}$	K_3-ded(19)
(21)	1	$\vdash$	$(\neg K_1p_1)_{w2}$	K_2-ded(20)
(22)	1	$\vdash$	$(K_3\neg K_2p_2)_{w0}$	Rulep(18)
(23)	1	$\vdash$	$(\neg K_2p_2)_{w1}$	K_3-ded(22)
(24)	1	$\vdash$	$(p_2)_{w2}$	K_2-gen(23)

These lines represent a stage of the proof about half way through. It remains to justify the right branch of the argument by cases (line 30).

(28)	28	$\vdash$	$(K_2\neg p_3)_{w1}$	Hyp
(29)	28	$\vdash$	$(\neg p_3)_{w2}$	K_1-ded
(30)	28, 1	$\vdash$	$(p_3)_{w1}$	NJ

The lines below represent the lowest closed branch of the cases argument.

(25)	25	$\vdash$	$(K_2p_3)_{w1}$	Hyp
(26)	25	$\vdash$	$(p_3)_{w1}$	K_1-ded
(27)	25, 1	$\vdash$	$(p_3)_{w1}$	Rulep(26)
(31)	1	$\vdash$	$(p_3)_{w1}$	cases: 16,28,25
(32)	1	$\vdash$	$(K_3p_3)_{w0}$	K_3-gen
(33)		$\vdash$	$((1 \wedge 2 \wedge 3 \wedge 4 \wedge 5) \supset K_3p_3)_{w0}$	deduct(32)

4.4 Two More Types of Outline Transformations

Thus far, all the material presented in this section has been directly justifiable in terms of a cut-free *LKM* proof. A "real" mathematician, however, would seldom produce a proof which is, in a sense, cut-free. Instead he makes suitable definitions and states and proves lemmata which build to the ultimate conclusion he seeks. Proofs stated in this form are naturally clearer and easier to read as they are shorter and more compact.

The method of outline transformations is a computational method of incrementally constructing a proof under partial or total guidance by a human. In order to make this method more palatable, we should allow the use of definitions and lemmata in this system.

The following outline transformations are one possible approach to the use of these tools. The introduction of lemmata and axioms leads to the need to use the cut rule. The use of definitions requires the modification of LKM to support definition inferences. Here, we must also assume some database axioms, definitions, and previously proved theorems of the system. In the sequel σ and ψ are arbitrary ornamented formulas.

P-Ax
This transform allows the introduction of external information from some external database. It is assumed that σ is a member of that database. The justification MP stands for modus ponens. In this case, $\Sigma' := \Sigma_0 \cup \{\ \Re; a, \Gamma_z \ \longrightarrow\ y\}$

(z)	$\mathcal{H}$	$\vdash$	ψ	NJ

$\Longrightarrow$

(a)	a	$\vdash$	σ	Given
(y)	$\mathcal{H}, a$	$\vdash$	ψ	NJ
(z)	$\mathcal{H}$	$\vdash$	ψ	MP

D-Def
This introduces definitions in supporting lines. ψ is an instance of σ where some subformula has been replaced by its definition. Replace a by b wherever it occurs in Σ to get Σ'.

(a)	$\mathcal{H}$	$\vdash$	σ	RuleX

$\Longrightarrow$

(b)	$\mathcal{H}$	$\vdash$	ψ	Def: a

P-Def
This transform allows the introduction of definitions in sponsoring lines. In the following, ψ is an instance of σ where some subformula of σ has been replaced by its definition. In this case, $\Sigma' := \Sigma_0 \cup \{\ \Re; \Gamma_z \ \longrightarrow\ y\}$

(z)	$\mathcal{H}$	$\vdash$	σ	NJ

$\Longrightarrow$

(y)	$\mathcal{H}, a$	$\vdash$	ψ	NJ
(z)	$\mathcal{H}$	$\vdash$	σ	Def:y

The justification for P-Ax is:

$$\frac{\Re; \Gamma \longrightarrow \sigma \qquad \Re; \Gamma, \sigma \longrightarrow \psi}{\Re; \Gamma \longrightarrow \psi} \text{ cut}$$

For definitions, we assume that the system LKM has been augmented with *definition inference rules.* By this, we mean the upper sequent of the rule differs from the lower by replacement of a subformula of a member of the lower sequent by its definition. The justifications are then:

$$\frac{\Re; \Gamma \longrightarrow \psi}{\Re; \Gamma \longrightarrow \sigma} \text{ Def-is}$$

$$\frac{\Re; \psi, \Gamma \longrightarrow (C)_{w'}}{\Re; \sigma, \Gamma \longrightarrow (C)_{w'}} \text{ Def-ia}$$

The example sessions shown in the appendix demonstrate the use of definitions.

5 Towards a Unification of Proof Representations

Thus far we have presented two proof systems. First, we introduced the LKM family. This system had the character of standard logical systems. This made it easy to discuss in a formal manner and demonstrate its properties. At the other end of the spectrum, the outline technique is clearly oriented toward computational editting of proofs. This makes the formalization straightforward to implement, but difficult to analyze.

Ideally, what we would like is some structure which is theoretically "clean", yet is also computationally oriented. If done correctly, it should be straightforward to translate this structure into LKM, outline transformations, or any other style of modal proof we might develop. The purpose of this section is to develop such a proof representation. This representation will be based on *expansion trees* as developed by Miller [18,19,21].

Expansion trees (ET's) were originally developed as a proof representation for higher order logic. As such, they provide a Herbrand type of result for that class of languages. ET-proofs are structured to be are a compact representation of a proof, yet contain all the essential information to generate Gentzen proofs, linear natural deduction proofs, Hilbert style proofs, and linear reasoning proofs [21].

As mentioned above, modal logics and higher order logics can be viewed as somewhat orthogonal extensions to classical first order logic. Since ET's are sufficient for higher order logic, the question naturally arises as to whether this structure is adaptible to modal logics. This section addresses this question by developing a modal analog of ET proofs, called *modal expansion tree proofs* (MET-proofs), which will encode a (propositional) modal proof.

We will motivate the discussion by examining the ET proof for the first order translation[14] of a formula in the system $S4$. This examination of ET's for modal translations will serve the dual purpose of providing an introduction to ET's for those unfamiliar with the formalism, and revealing certain structural regularities which will motivate the definition of MET's. The result will be a structure resem-

[14]See appendix section B for an explanation of the translation of modal statements to first order equivalent statements

bling ET's and the criteria for that structure which deliniate proofs from non-proofs of modal statements.

5.1 MET Proofs

An expansion tree represents both the logical structure of a formula, and the substitutions required to generate a tautologous instance of it. Two functions, Sh and Dp, map expansion trees to a quantified formula and to the instantiated form of the formula respectively. Rather than use Skolemization to ensure proper use of substitution, expansion trees have two order relations defined on the substitution terms which serve the same purpose.

The following definition is a slightly modified version of that found in [21]. In the sequel, let $\mathbf{S}^x_t A$ denote A with the substitution of t for all free occurrences of the variable x in A.

Definition 5.1 We now define *expansion trees, dual expansion trees, selected variables, expansion terms*, and two functions Sh and Dp which map expansion trees and dual expansion trees to formulas.

1. If A is an atom, p, then A is both an expansion tree and a dual expansion tree. $Dp(A) := Sh(A) := p$.
2. If Q is an expansion tree (dual expansion tree), then $\sim Q$ is a dual expansion tree (expansion tree). $Sh(\sim Q) :=\sim Sh(Q)$ and $Dp(\sim Q) :=\sim Dp(Q)$.
3. If Q_1 and Q_2 are expansion trees or dual expansion trees, then so are $Q_1 \mathbin{\%} Q_2$. $Sh(Q_1 \mathbin{\%} Q_2) := Sh(Q_1) \mathbin{\%} Sh(Q_2)$ and $Dp(Q_1 \mathbin{\%} Q_2) := Dp(Q_1) \mathbin{\%} Dp(Q_2)$.
4. If $\mathcal{Q}xA$ is universal and occurs positively (existential and occurs negatively), and Q is an expansion tree (dual expansion tree) for $\mathbf{S}^x_y A$, y not selected in Q, then $Q' := A +^y Q$ is an expansion tree (dual expansion tree). $Sh(Q') := A$ and $Dp(Q') := Dp(Q)$. We say y is *selected* in Q'.
5. If $\mathcal{Q}xA$ is existential and occurs positively (universal and occurs negatively), and $Q_1, \ldots Q_n$ is a list of expansion trees (dual expansion trees) for $\mathbf{S}^{x_1}_{t_1} A, \ldots, \mathbf{S}^{x_n}_{t_n} A$ then $Q' := A +^{t_1} Q_1 +^{t_2} \ldots +^{t_n} Q_n$ is an expansion tree (dual expansion tree) with expansion terms $t_1, \ldots t_n$. $Sh(Q') := A$ and $Dp(Q') := Dp(Q_1) \vee \ldots \vee Dp(Q_n)$ $(Dp(Q') := Dp(Q_1) \wedge \ldots \wedge Dp(Q_n))$.

We will say an arc or node *dominates* another arc or node in and MET if the former occurs above the latter in that tree. We will also use the symbol $\supset$ as an abbreviation for $\sim \ldots \vee \ldots$.

Let S_Q and Θ_Q be the set of selection and expansion terms respectively for an ET, Q.

Definition 5.2 Let $<^0_Q$ be a binary relation on S_Q such that $z <^0_Q y$ if there is a $t \in \Theta_Q$ such that z is free in t and a node dominated by (the arc labelled with) t is selected by y. Let $<_Q$ denote the transitive closure of $<^0_Q$. This is called the *imbedding* relation for Q.

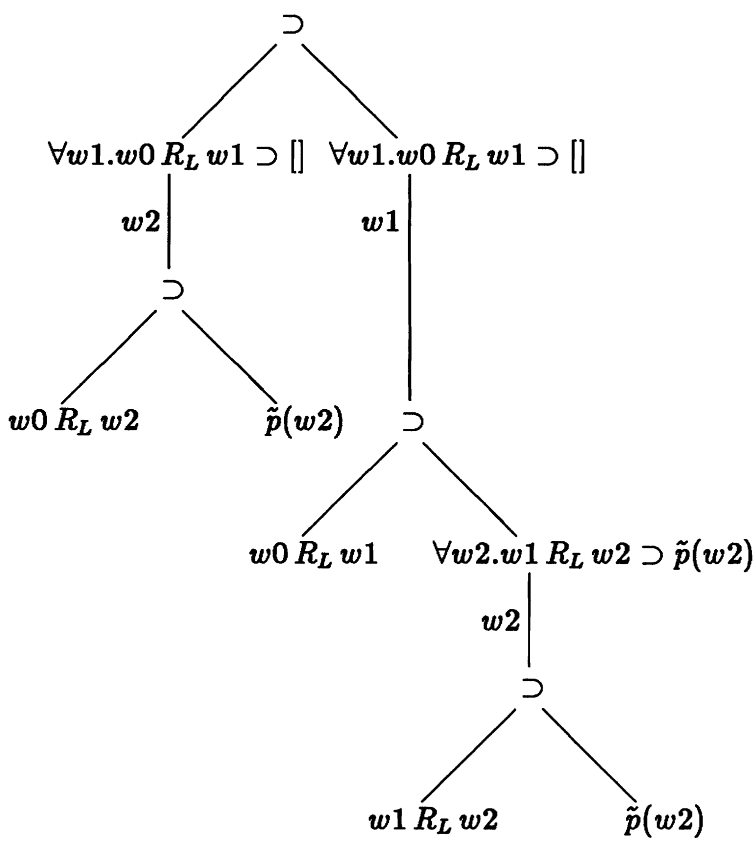
Definition 5.3 An expansion tree, Q is *sound* if none of the free variables in $Sh(Q)$ are selected in Q. An expansion tree is an expansion tree *for* A if $Sh(Q) = A$ and Q sound. An *ET-proof* for A is an expansion tree, Q, for A such that:

1. $Dp(Q)$ is tautologous, and
2. $<_Q$ is acyclic.

Figure 1 is an ET for the translation of $Lp \supset LLp$. This ET can be turned into an ET-proof by attaching, by an implication operator, another ET representing the possible worlds relation constraints such that the deep form of the entire structure is tautologous.

Expansion trees for the translations of modal formula would be a sound and complete proof representation for modal systems of logic. However, we can make the following observations. Note that each occurrence of a modality appears in the tree as a selection or expansion followed by an implication operator. This makes the ETs very "bushy" compared to the information they hold. Moreover, in the first-order case, these selections and expansions treat quantification over worlds the same as quantification over individuals. We generally would like to differentiate between worlds and individuals. Lastly, every expansion tree will be prefaced by the restrictions on the possible worlds relation which are all of essentially the same form.

These observations lead us to define a new structure, *modal expansion trees*, which are a specialization of expansion trees to modal logic. METs will resemble ETs, but are made more compact by distinguishing between relational information and propositional information. In particular, we will add *generation* and *deduction* nodes which mirror selection and expansion nodes, respectively, but are only used for the possible worlds relation.



$Sh(\mathcal{M}) = [\forall w1 \,.w0\, R_L\, w1 \supset \tilde{p}(w1)] \supset \forall w1 \,.w0\, R_L\, w1 \supset \forall w2 \,.w1\, R_L\, w2 \supset \tilde{p}(w2)$

$Dp(\mathcal{M}) = [w0\, R_L\, w2 \supset \tilde{p}(w2)] \supset [w0\, R_L\, w1 \supset .w1\, R_L\, w2 \supset \tilde{p}(w2)]$

$S = \{w1,\, w2\}$

$\Theta = \{w2\}$

$<_{\mathcal{M}} = \{\}$

Figure 1: An ET for the translation of $Lp \supset LLp$

Definition 5.4 Here we define the *modal expansion tree* (MET) of a formula A, denoted $\mathcal{M}$, the functions Dp and Sh, *generation* and *deduction* variables, *associated* literals, and *free world variable*.

1. If A is an atom, p, then $Q := (p)_w$ is both an expansion tree and a dual expansion tree. $Dp(Q) := \tilde{p}(w)$ and $Sh(Q) := (p)_w$. We will refer to w as the *free world variable* in Q and denote this by $Q[w]$ We will often refer to a tree of this form as the *trivial tree*, and $\tilde{p}(w)$ as the literal *associated* with this node.
2. If $Q[w]$ is an expansion tree (dual expansion tree), then $(\sim Q)[w]$ is a dual expansion tree (expansion tree). $Sh(\sim (Q)_w) := (\sim Sh(Q))_w$ and $Dp(\sim Q) :=\sim Dp(Q)$.
3. If $Q_1[w]$ and $Q_2[w]$ are expansion trees or dual expansion trees, then so is $(Q_1 \mathbin{\wedge\!\!\!\vee} Q_2)[w]$. If $Sh(Q_1) = \alpha_w$ and $Sh(Q_2) = \beta_w$, then $Sh(Q_1 \mathbin{\wedge\!\!\!\vee} Q_2) := (\alpha \mathbin{\wedge\!\!\!\vee} \beta)_w$, and $Dp(Q_1 \mathbin{\wedge\!\!\!\vee} Q_2) := Dp(Q_1) \mathbin{\wedge\!\!\!\vee} Dp(Q_2)$.
4. If A is universal and occurs positively (existential and occurs negatively), the matrix of A is A', the operator of A is ρ and $Q[w]$ is an expansion tree (dual expansion tree) for A', then $Q' := (A +^w Q)[x]$ is an expansion tree (dual expansion tree) provided x is not generated in Q. $Sh(Q') := (A)_x$ and $Dp(Q') := x\,R_\rho\,w \supset Dp(Q)$ $(Dp(Q') := x\,R_\rho\,w \wedge Dp(Q))$. w is the *generation variable* for Q', and $x\,R_\rho\,w$ is the literal *associated* with this generation arc.
5. If A is existential and occurs positively (universal and occurs negatively), the matrix of A is A', the operator of A is ρ and $Q_1[w_1], \ldots Q_n[w_n]$ is a list of expansion trees (dual expansion trees) for A' then $Q' := (A +^{w_1} Q_1 +^{w_2} \ldots +^{w_n} Q_n)[x]$ is an expansion tree (dual expansion tree) with *deduction terms* $w_1, \ldots w_n$. $Sh(Q') := (A)_x$ and $Dp(Q') := (x\,R_\rho\,w_1) \wedge Dp(Q_1)) \vee \ldots \vee (x\,R_\rho\,w_n \wedge Dp(Q_n))$ $(Dp(Q') := (x\,R_\rho\,w_1 \supset Dp(Q_1)) \wedge \ldots \wedge (x\,R_\rho\,w_n \supset Dp(Q_n)))$. $x\,R_\rho\,w_i$ is the literal *associated* with the deduction arc labelled with w_i.

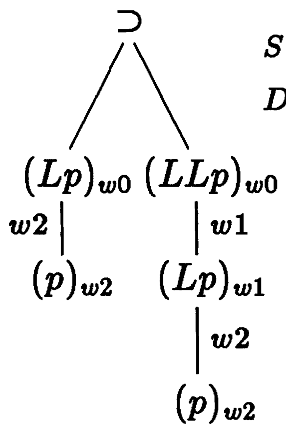
Generation and deduction terms are a restricted case of selection variables and expansion terms, and so must satisfy many of the same properties. In particular, they have a similar order relation.

Definition 5.5 Given an MET, $\mathcal{M}$, let

$$\Sigma_{\mathcal{M}} = \{w | w \text{ is a deduction variable in} \mathcal{M}\}$$

$$\Upsilon_{\mathcal{M}} = \{w | w \text{ is an generation variable in } \mathcal{M}\}$$

Definition 5.6 Let $\mathcal{M}$ be an MET. Define a relation $<^0_{\mathcal{M}}$ on Υ by $w_i <^0_{\mathcal{M}} w_j$ if $w_i \in \Sigma$, and the arc labelled with w_i dominates the arc labelled with w_j. Let $<_{\mathcal{M}}$ denote the transitive closure of $<^0_{\mathcal{M}}$.



$$Sh(\mathcal{M}) = Lp \supset LLp$$

$$Dp(\mathcal{M}) = [w0\, R_L\, w2 \supset \tilde{p}(w2)] \supset [w0\, R_L\, w1 \supset .w1\, R_L\, w2 \supset \tilde{p}(w2)]$$

$$\Sigma = \{w2\}$$

$$\Upsilon = \{w1,\ w2\}$$

$$<_{\mathcal{M}} = \{\}$$

Figure 2: The MET proof for $Lp \supset LLp$

The MET, as defined, represents the structure of the modal formula and some of the possible worlds structure. We must somehow represent the theory of the possible worlds relation. In the most general case, we will do this by associating and expansion tree for the possible worlds theory with certain conditions attached. For theories such as T or $S4$, we can in fact collapse the structure of MET's to be much more concise.

Definition 5.7 An MET $\mathcal{M}$ of A is an MET *proof* for A if

1. There is a sound expansion tree, Q, such that $Dp(Q) \supset Dp(\mathcal{M})$ is tautologous,
2. The free variable of $\mathcal{M}$ does not appear generated in $\mathcal{M}$,
3. No generated world variable of $\mathcal{M}$ appears selected in Q, and
4. $<_{\mathcal{M}} \cup <_{Q}$ is acyclic[15].

We can derive different classes of MET's by restricting Q to be an expansion tree for some set of axioms specifying the restrictions on the possible worlds relation. For example, figure 5.1 is the MET-proof for $Lp \supset LLp$ in $S4$. Notice how this representation compares with that of figure 1.

We will show METs are sound and complete by demonstrating equivalence of the MET for a formula to the ET of the first order translation of a formula.

Proposition 5.1 A modal formula M has an MET proof, $\mathcal{M}$, iff it's translation, A, has an ET proof.

[15]We will abuse notation and henceforth denote this as $<_{\mathcal{M}\supset Q}$.

Proof: (sketch)

Proof of this proposition involves showing that we can convert one proof representation to the other and still yield a proof.

1. Assume we have an MET proof. By the previous discussion, there is a simple recursive translation from an MET proof to a sound ET, Q_2, of $Sh(\mathcal{M})$. Thus, the deep formulas are logically equivalent. By definition, there is some ET Q_1 such that $Q_1 \supset Q_2$ has a tautologous deep formula. Imbedding relation of this ET is identical to that of the MET, and so is acyclic. Hence, this is an ET-proof.
2. The proof in the other direction is similar, and so is omitted.

♣

Corollary 5.2 MET's are a sound and complete proof representation for modal logic.

Proof: By virtue of the preceding proposition, the correctness of translation to first-order equivalent statements, and the soundness and completeness of ET's as a proof representation. ♣

5.2 Automatic Generation of *LKM* proofs

There is a striking similarity between MET-proofs and sequential proofs as presented in section 3. We can, in fact, show an explicit connection between MET-proofs and sequential proofs by demonstrating a procedure for *translating* MET-proofs to sequential proofs. This procedure will also provide an example of the use of the various properties of MET's, and provides, as a corollary, a completeness result for *LKM without* the cut rule.

The approach will be to define an analog sequent system which operates on MET-proofs. This system will be in exact correspondance with the *LKM* system, and proofs in the analog system will be directly translatable in to *LKM* proofs via use of *Sh* as defined over MET's. Following the terminology of Miller [20], we shall call this the *Q-system* of inference; and, it shall contain *Q-sequents* and *Q-inference rules.*

Definition 5.8 An *Q-sequent* is a structure of the form:

$$Q; N_1, \ldots, N_r \longrightarrow M_1, \ldots, M_s \text{ where}$$

1. N_i is a dual modal expansion tree
2. M_i is a modal expansion tree
3. Q is a sound expansion tree
4. Q and $\sim N_1 \wedge \ldots \wedge \sim N_r \vee M_1 \vee \ldots \vee M_s$ comprise an MET-proof.

We shall say that the sequent:

$$Sh(Q); Sh(N_1), \ldots, Sh(N_r) \longrightarrow Sh(M_1), \ldots, Sh(M_s)$$

is the LKM sequent ***associated*** with this Q-sequent.

For the purposes of this construction, consider LKM with the structural inference rules Thinning and Contraction, propositional inference rules $\vee$–IA, $\vee$–IS, $\neg$–IA, $\neg$–IS, and modal inference figures L–IA and L–IS. We will assume that sequents are multisets, negating the need for interchange. In the sequel, we will refer to a world x as ***admissible*** in a Q-sequent if x does not appear in Υ associated with that Q-sequent.

We can now define several Q-inference figures analogous to the LKM inference figures listed above.

Definition 5.9 (The Q-analog Inference Figures)
Here, A and C are METs, and Γ and Θ are sets of METs.

$$\frac{Q; A[w], \Gamma \longrightarrow \Theta \qquad\qquad Q; C[w], \Gamma \longrightarrow \Theta}{Q; (A \vee C)[w], \Gamma \longrightarrow \Theta} \quad \vee\text{–IA}_Q$$

$$\frac{Q; \Gamma \longrightarrow \Theta, A[w]}{Q; \Gamma \longrightarrow \Theta, (A \vee C)[w]} \quad \vee\text{–IS}_Q \qquad \frac{Q; \Gamma \longrightarrow \Theta, (C)[w]}{Q; \Gamma \longrightarrow \Theta, (A \vee C)[w]} \quad \vee\text{–IS}_Q$$

$$\frac{Q; \Gamma \longrightarrow \Theta, A[w]}{Q; (\neg A)[w], \Gamma \longrightarrow \Theta} \quad \neg\text{–IA}_Q \qquad \frac{Q; A[w], \Gamma \longrightarrow \Theta}{Q; \Gamma \longrightarrow \Theta, (\neg A)[w]} \quad \neg\text{–IS}_Q$$

$$\frac{Q; (Lp +^{t_1} p_1 + \ldots +^{t_{i-1}} p_i - 1 +^{t_{i+1}} \ldots +^{t_n} p_n)[w], p_i[t_i], \Sigma \longrightarrow \Theta}{Q; (Lp +^{t_1} p_1 + \ldots +^{t_i} p_i + \ldots +^{t_n} p_n)[w], \Sigma \longrightarrow \Theta} \quad L\text{–IA}_Q{*}^{\dagger}$$

$$\frac{Q; p[x], \Sigma \longrightarrow \Theta}{Q; (Lp +^{x} p)[w], \Sigma \longrightarrow \Theta} \quad L\text{–IA}_Q{}^{\dagger} \qquad {}^{\dagger}\frac{Q, w\, R_L\, x; \Sigma \longrightarrow p[x], \Theta}{Q; \Sigma \longrightarrow Lp[w], \Theta} \quad L\text{–IS}_Q$$

†Note that t_i and x must be admissible.

Proposition 5.3 If the lower structure of the inference figures of definition 5.9 is a Q-sequent, the upper structure is a Q-sequent.

Proof:

First, observe that modal expansion trees and dual modal expansion trees are preserved under the structural changes given in the inference rules. If the imbedding relation was acyclic in the lower Q-sequent, it is in the upper Q-sequent. Moreover, in each case, Q is unchanged, or has an unquantified atom attached via a conjunct. These changes sustain Q's status as a sound expansion tree.

Lastly, for the propositional rules it is straightforward to verify that the tautologous nature of the deep forms holds. It remains to shown that the tautology condition is preserved by the modal inference figures.

For $L-$IS, note that Dp of the lower stucture is of the form

$$Dp(Q) \supset \sim Dp(\Theta) \vee (w\, R_L\, x \supset \tilde{p}) \vee Dp(\Gamma)$$

which is equivalent to

$$Dp(Q) \wedge w\, R_L\, x \supset \sim Dp(\Theta) \vee \tilde{p} \vee Dp(\Gamma)$$

This is the deep form for the upper sequent.

For $L-$IA, the lower structure has the deep form:

$$Dp(Q) \supset \sim Dp(\Theta) \vee \sim (w\, R_L\, x \supset \tilde{p}) \vee Dp(\Gamma)$$

which is tautologous iff

$$Dp(Q) \supset w\, R_L\, x$$

and

$$Dp(Q) \supset \sim Dp(\Theta) \vee \sim \tilde{p} \vee Dp(\Gamma)$$

are tautologous.

The latter is the deep form for the upper structure. Note that the former statement corresponds to the proviso on $L-$IA, while the admissibility condition corresponds to the proviso of $L-$IS.

♣

We have now verified that the inference rules are properly defined. It is simple to show that one of these inference rules is always applicable until the members of the Q-sequent consist only of trivial trees. Merely observe that the propositional rules are always applicable, and the acyclic nature of $<$ ensures that some deduction term is always eliminable.

Notice that the tautologous nature of the deep form ensures that there must be matching atoms in the expansion trees. That is, if we take Sh of a leaf sequent consisting of only trivial trees, then it must be derived from an LKM axiom soley through applications of thinning.

Proposition 5.4 If $\mathcal{M}$ is a MET-proof for A, then there is a corresponding (cut-free) LKM proof for A.

> **Proof:** Consider a Q-inference which has $\mathcal{M}$ at the root, and Q-sequents consisting only of trivial trees at the leaves. For any rule but L—IA$*$, replace the sequents by the Sh form and drop the Q subscript. For example, for $\vee$—IA we have

$$\frac{Q; Sh(A[w]), Sh(\Gamma) \longrightarrow Sh(\Theta) \qquad Q; Sh(C[w]), Sh(\Gamma) \longrightarrow Sh(\Theta)}{Q; Sh((A \vee C)[w]), Sh(\Gamma) \longrightarrow Sh(\Theta)} \vee\text{—IA}_Q$$

> For L—IA$_Q*$, add the following proof segment:

$$\cfrac{\cfrac{Sh((Lp +^{t_1} p_1 \ldots)[w]), Sh(p_i[t_i]), Sh(\Gamma) \longrightarrow Sh(\Theta)}{Sh((Lp +^{t_1} p_1 \ldots)[w]), Sh((Lp +^{t_1} p_1 \ldots)[w]), Sh(\Gamma) \longrightarrow Sh(\Theta)} L\text{—IA}}{Sh((Lp +^{t_1} p_1 \ldots)[w]), Sh(\Gamma) \longrightarrow Sh(\Theta)} \text{contraction}$$

> Finally, apply the appropriate Thinning's to the leaf sequents of the partial LKM tree to derive LKM axioms. This final object will be an LKM proof for A.
>
> ♣

5.3 Automatic Generation of Natural Deduction Proofs

As an example of the use of MET proofs, and also as a (relative) completeness result of the outline transformations, we will sketch how to convert an MET proof into a natural deduction proof. The algorithm we will present is highly non-deterministic and is only a starting point. There are several changes that could be incorporated to both increase the efficiency of the process, and to make the proofs more "elegant".

The crux of the method used to generate natural deduction proofs from METs is that, by associating a portion of the MET with each line of the proof, we can use the information present to select outline transformations, and substitute the proper generation and deduction terms where appropriate. To be completely formal in our argument, we should redo Definition 4.1 to accomodate the MET associated with each line. Then, we would need to add the MET manipulations to each of the outline transforms. Most of these manipulations are straightforward, however, and due to time and space restrictions will not be given. We shall instead provide a purely informal account. The interested reader should consult [18] for the complete account.

There are two fundamental results we need about proof outlines.

Proposition 5.5 If $\mathcal{O}$ is an outline which contains a non-atomic active lines, then some outline transformation can be applied to $\mathcal{O}$.

Proof of this proposition rests on the fact that we can either eliminate a standard connective, or (based the acyclic nature of $<$ defined on the MET associated with this line) apply a transformation for a modality.

Proposition 5.6 If $\mathcal{O}$ is an outline in which all active lines assert atoms, then RuleP1 or RuleP2 can be applied to the outline for all lines needing justification.

Proof of this proposition is based on that fact that the deep formula would not be tautologous is this were not the case.

In the ensuing discussion, we will assume we have a spanning mating for the deep formula.

Proposition 5.7 If l is a proof line in $\mathcal{O}$, and no literal of the MET associated with l is mated, then D-Thinning or P-thinning (whichever is appropriate) can be applied and it will not interfere with the closure of $\mathcal{O}$.

Justification of this proposition is based on that fact that the paths of the sequents associated with this outline do not use the literals in this particular formula. Hence, it can be removed without changing the essential structure of the outline.

An algorithm for generating a natural deduction proof for a formula, given an MET for that formula, is:

1. Initially set $\mathcal{O} := \mathcal{O}_0$
2. Apply P-Neg or D-Neg to any lines which assert top level negations.
3. Apply some outline transformation to $\mathcal{O}$
4. Apply D-thinning or P-thinning to any applicable proof line
5. If some active line is not atomic, go to step 2.
6. Apply RuleP1 or RuleP2 to all active sequents to close the outline.

The three preceding propositions assure the algorithm is correct and will always yield a proof given an MET. We must, however, ensure it terminates. Such a proof is based on the following observations:

1. Any application of a transformation from the base set reduces the complexity of some line in the proof, deactivating that line in the process.
2. Any applications of a modality transformation to some line introduces at most a finite set of lines.
3. Any application of a RuleP variant closes some line and removes a sequent.

Based on these observations, we see that the inner loop of the algorithm will eventually terminate (if not, we would need have infinite formula, or and infinite expansion tree, both of which are disallowed). The application of RuleP rules must also terminate. Thus, the algorithm terminates.

5.4 Automatic Generation of MET-proofs

We will now explore some aspects of the actual generation of MET-proofs. The generation will be based on proving the validity of Dp of the modal expansion tree for some formula. It would be possible, of course, to use tableau methods as the basis for generation. However, there are some reasons why first-order methods may be appropriate. First, very efficient first-order theorem provers exist, and semantic translations for numerous modal logics are first-order. Thus, it is time-efficient to add a translator to the front end of a theorem prover, rather than develop a new theorem prover. Second, it is not entirely clear that tableau methods are applicable to first order modal theories, while there are translations which account for first order modal logic. Thus, we will consider first-order methods. In particular, we prove some properties about the search space for MET-proofs which enhance the efficiency of the proving process.

We can view the search for an MET-proof as a two-stage process. First, construct an MET for the formula of interest and then search for a substitution which makes deep structure tautologous under the constraints imposed by the possible worlds relation, and has an acyclic imbedding relation. Practically, we use skolemization to encode the imbedding relation and use standard theorem provers to check for validity of the translation of the shallow formula, giving us the proper instantiation for the selection and expansion arcs as a by-product. It is simple to convert a skolemized tree to the structure we have defined.

The structure of the formulae which we are considering is fairly rigid. The question arises as to whether we must examine the full search space in the course of theorem proving. As it turns out, there are sound and complete heuristics for pruning the search space. We will present those heuristics based on analysis of the method of matings [2]. We will assume the reader has some familiarity with this method.

To quickly summarize the process, a first-order formula in nnf is tautologous if and only if it has a mated pair of literals along every *path* in the formula.[16] Paths are defined over formulas in *negation normal form*, and correspond roughly to the clauses of the formula in conjunctive normal form. A pair of atoms on a path is mated if there is some substitution such that the atoms are made identical under that substitution. By analysis of the paths of the deep formula, we will see that there is an intelligent way to search for these matings.

[16]Rather than supply a formal definition, we refer the reader to the appendix, Section A, page 82 where the Prolog theorem prover is listed. The definition of path in that code is essentially the formal definition of path.

In the sequel, we will assume a unique labelling of the literals[17] of a formula of the form $r_1, r_2, \ldots$ for relational atoms (r-literals in the sequel), and $l_1, l_2, \ldots$ for other atoms (p-literals in the sequel). We will also denote the set of paths for the deep formula of an MET by Π. It will be convenient to distinguish between r-literals and p-literals on paths, so we view paths a pairs of sets, e.g. $\langle (r_1, \ldots, r_m), (l_1, \ldots, l_n) \rangle$. A *mated pair* will be a tuple $\langle p_1, p_2 \rangle$ such that p_1 and p_2 are complementary. A mated pair $\langle p_1, p_2 \rangle$ is a *mating* for a path $\langle R, P \rangle$ if either R or P contains both p_1 and p_2. We will often refer to matings of r-literal as relational matings, and other matings as propositional matings. A set of mated pairs is a *spanning mating* for a set of paths if and only if each path has a mating from that set. For the purposes of the following discussion, we will call a literal occurrence *essential* with respect to a some mating if it appears as a member of a mated pair. Finally we will also assume (for convenience sake) that all logics have both the positive and negative sense of the modalities.

Definition 5.10 A modal formula containing both L and M modalities is in *negation normal form (nnf)* if the scope of each negation operator extends to only atomic propositions.

Henceforth, we will only consider modal formulae in nnf. It is simple to show that all propositional modal statements have an equivalent nnf. We immediately observe the following:

Proposition 5.8 If A is a modal formula in nnf, and $\mathcal{M}$ is an MET for A, then both $\tilde{A}$ and $Dp(\mathcal{M})$ are in first-order nnf (modulo rewrite of the newly introduced implication signs).

> **Proof:** We show by induction that the deep formula of an MET for a formula in nnf results in a first order formula in nnf. The proof for $\tilde{A}$ is similar and left for the reader.
>
> - If A is a literal or the negation of a literal then $Dp(\mathcal{M})$ is in nnf.
> - If A is $A_1 \between A_2$ then, by the induction hypothesis, $Dp(\mathcal{M}_{A_1})$ and $Dp(\mathcal{M}_{A_2})$ are in nnf, hence so is $Dp(\mathcal{M})$
> - If A is $L(A')$, then $Dp(\mathcal{M}_A)$ is $x\, R_\rho\, w \supset Dp(\mathcal{M}_{A'})$ which rewrites to $\sim x\, R_\rho\, w \vee Dp(\mathcal{M}_{A'})$. The negation extends only to a literal, and $Dp(\mathcal{M}_{A'})$ is in nnf by hypothesis.

[17] We take a literal to be an atom, or the negation of an atom.

- If A is $M(A')$, then $Dp(\mathcal{M}_A)$ is $(x\, R_\rho\, w_1 \wedge \tilde{Dp}(\mathcal{M}^1_{A'}[w_1]) \vee \ldots \vee (x\, R_\rho\, w_n \wedge \tilde{Dp}(\mathcal{M}^n_{A'}[w_n])$ which, by appealing to the induction hypothesis, is in nnf.

♣

As we said at the beginning, there is a substantial amount of structure associated with the deep formula of an MET in nnf. In order to expose this structure, we need to define *dominance* as a relation between positive r-literals (which are associated with deduction nodes) and literals.

Definition 5.11 We will say a positive r-literal, r, *dominates* another literal, l, in an MET if the deduction arc associated with r occurs above the node or arc associated with l in the MET. We will call a sequence of r-literals $r_1, r_2, \ldots, r_n$ a *dominating chain* of l in an MET if r_1 dominates l, r_i dominates r_j for $i < j$ and there is no positive r-literal which dominates l and does not appear in the chain.

By virtue of the special structure of the class of MET's, we can state the following fundamental proposition.

Proposition 5.9 If a path $\langle R, P\rangle \in \Pi$ contains a literal dominated by an r-literal, r, then there is a path $\langle R', P'\rangle \in \Pi$ where:

$$R' = R - \{r' | r' \text{is dominated by } r\}$$

$$P' = P - \{p' | p' \text{ is dominated by } r\}$$

We will often write $\langle R, P\rangle \prec_r \langle R', P'\rangle$ to denote this relationship between paths.

Proof:

This can easily be seen by looking at structure of the deep formula of the MET. Positively occurring relation literals occur *only* in deduction nodes. Thus, they are attached by a conjunctive operator when the formula appears in nnf. Hence, when constructing paths for the deep formula, there will be a bifurcation; half of the paths will contain the r-literal, and half will contain the path extensions generated by the subtree conjoined to the r-literal.

♣

Definition 5.12 Let $\Pi^0 = \{\pi | \pi \in \Pi \text{ contains no positive } r\}$

Proposition 5.10 For any modal formula, $\Pi^0 \neq \emptyset$

Proof:

By the fact that MET's are non-empty and finite, by repeated application of the Proposition 5.9 we can arrive at a path with no positive r-literals.

♣

It is easily seen that all elements of Π^0 are minimal with respect to $\prec$. Another simple consequence of $\prec$ is the following:

Proposition 5.11 If $\pi \in \Pi$ contains a mated pair of p-literals, $\langle l_1, l_2 \rangle$, then $\langle l_1, l_2 \rangle$ is a mating for all $\pi' \in \Pi$ such that $\pi' \prec \pi$.

Proof:

This is, again, a very simple result based on the fact that the π' differ only by the replacement of relation propositions with more literals. Hence, if some mated pair was a mating for π, both members of that pair appear in π' and hence that mated pair is a mating for π'. [18] ♣

Note the existence of Π^0 implies a result analogous to the standard result in modal logic (e.g [11] pg. 41) which states that the *PC-transform* of a modal theorem is tautologous. These paths correspond to the propositional part of the formula. If there is a spanning mating for Π, then the only matings for these paths *must* be propositional. Hence, the PC-transform is tautologous.

Also, note the following property of paths. If there are π and π' in Π where $\pi \prec_r \pi'$, and r dominates an essential literal in π, then the mating for π' *must* occur within the relational portion of the path, or be an alternate propositional mating for π by Proposition 5.11.

It is convenient to view the set of paths containing a dominating literal of some member of Π^0 as a lattice-like structure.[19] For the simplest case, referring to figure 5.4, assume we have one dominating r-literal, r_1, in an MET. The lattice for this tree has two elements. Assume there is a spanning mating for these paths. Then the least element has a propositional mating. If this is not a mating for the greatest element, then r_1 must dominate an essential literal. Note, since r_1 is the only positive r-literal, it immediately follows that r_1 must be mated.

[18] This is, in fact, a particularization of a more general result stating that a mating for some path is a mating for all paths which are extensions of the mated one.

[19] They are in fact a lattice, but we make no use of formal lattic properties here.

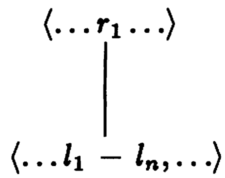


Figure 3: A Simple Example of a Path Lattice

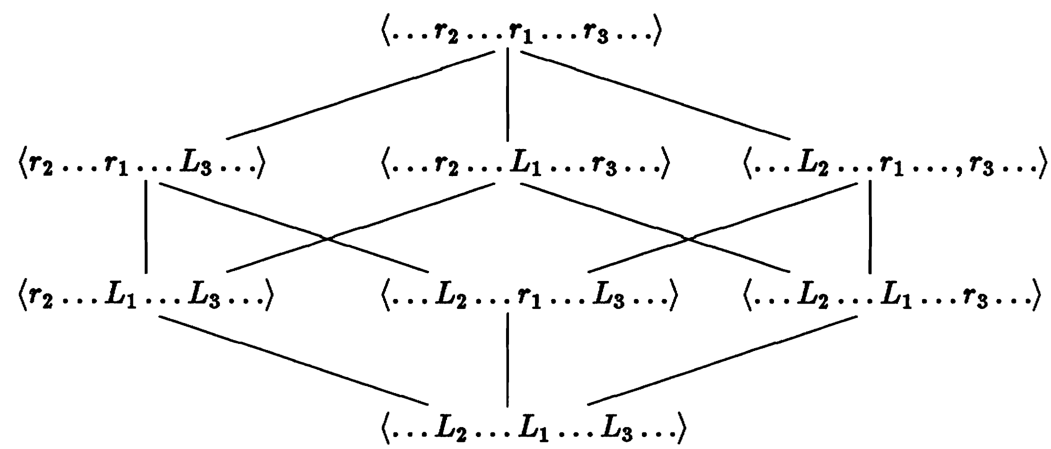


Figure 4: A More Complex Example of a Path Lattice

Suppose we have three relation literals, r_1, r_2, r_3, which dominate sets of literals L_1, L_2, L_3 (see figure 5.4). Furthermore, suppose that there are only two possible matings for the least element of the lattice. One mating uses only literals under r_2, and the other uses one literal dominated by r_1, and one dominated by r_3. If we look at all immediate predecessors of the least element, they can all achieve propositional matings. However, at the next level there is only one path which could have a propositional mating. The greatest element must have a relational mating. If r_2 was the mated relation literal, then r_2 and the propositional mating under it would suffice as a spanning mating for the entire lattice.

If r_2 is not mated, then look at the paths r_1 or r_3. In each case, we can find paths which, under the stated restrictions, force a mating for those literals. The sum of these matings is, in fact, a spanning mating for the entire lattice.

We would like to generalize these ideas and formalize the notion that certain r-literals are essential to a mating, and some are not. By doing this, we can examine exactly what structure the essential r-literals have. Intuitively, when we view an MET, it seems the only essential r-literals should be those that have some bearing on the propositional mating. In the examples this was true; only dominating r-literals of the propositionally mated pairs were important. We will formalize this by defining a *strong mating*.

Definition 5.13 A *constraint set*, C, for a path $\langle R, P\rangle$ and a positive r-literal r dominating at least one literal in the path is defined as:

$$C \subseteq \{r' | r' \in R \text{ and } r \text{ does not dominate } r'\}$$

We will refer to the pair $\langle r, C\rangle$ a *constraint* of the path $\langle R, P\rangle$.

Definition 5.14 A literal, l, in a path π has *satisfied constraints* if:

1. The are no r-literals dominating l (in which case there are no constraints), or
2. For all r dominating l, r is identical to a negative occurrence in the associated constraint set, and that literal has satisfied constraints.

Definition 5.15 A *strong mating* for a path, $\pi \in \Pi^0$ is a mating, $\langle l_1, l_2\rangle$ for π such that both l_1 and l_2 have satisfied constraints. A *strong spanning mating* is a strong mating for all members of Π^0.

A strong mating, then, consists of a propositional mating, matings for a dominating chain of the members of that propositional mating, and matings for members of the dominating chains of all r-literals mated to members of some dominating chain. We will now proceed to show that a strong spanning mating is a spanning mating, and that a spanning mating must include a strong spanning mating.

Proposition 5.12 If $\pi \in \Pi^0$, and $\pi \prec_r \pi'$, then r together with the negative r-literals in π form a constraint with respect to all literals dominated by r.

Proof: Directly from the definition of $\prec$ and constraint. ♣

Proposition 5.13 If all matings for a path, π, depend on at least one member of some subset $\pi' \subseteq \pi$, then at least one member of that subset has all constraints satisfied.

Proof:

We will show by strong induction that some member of the selected set must have all of its constraints satisfied.

The measure we will use is:

- 0 if some member of the set has no dominating r-literals.
- the maximum, over all members of the set, of the highest number of constraints which might need to be satisfied for that member.

1. If there are no dominating r-literals for some literal in π', then the result is trivial; there are no constraints for that literal.
2. Suppose that there are $n + 1$ constraints for some member. Consider the path $\langle R_1, P_1 \rangle$ where all members of the specified set are replaced by their immediately dominating r-literal as in Proposition 5.9. The only possible matings in this path must use these new positive r-literals. Now, consider $R_1{}' \subset R_1$ consisting of all negative r-literals which participate in a mating. Some member of this set has $\leq$ n constraints, and all matings depend on this set. By the induction hypothesis, at least one member has all constraints satisfied.

   Now, consider $R_1'' \subset R_1$ consisting of all positive r-literals participating in a mating with a r-literal having satisfied constraints and negative r-literals with no satisfied constraints. All matings depend on this set, some member of this has $\leq n$ constraints, and so has satisfied constraints. Moreover, this member must be mated to another r-literal with satisfied constraints, by the way R_2'' was constructed. Hence the original set has some member with satisfied constraints.

♣

Lemma 5.14 If some member of Π^0 has several matings in some spanning mating, the constraints of both members of one of those mated pairs are mated in that spanning mating.

Proof: Consider selecting one member of each mated pair and applying Proposition 5.13. At least one member of this set has satisfied constraints. We can symmetrically do this with the other member of all mated pairs. The question arises as to whether *both* members of some mated pair have satisfied constraints. Suppose not. Then each mated pair has at least one member which does not have satisfied constraints. Take all members of mated pairs which are unsatisfied. This is a set to which Proposition 5.13 can be applied, and hence some member must have satisfied constraints. This is a contradiction. Our assumption must have been wrong, and both members of some mated pair have satisfied constraints in this path. ♣

Proposition 5.15 If A has a spanning mating, then it has a strong spanning mating.

Proof: Every member of Π^0 has a mated pair in that spanning mating. By the previous proposition, some mated pair in each of those paths has satisfied constraints with respect to that path.

♣

Proposition 5.16 In a path set Π, A strong spanning mating for Π^0 of an MET-proof yields a spanning mating for Π.

Proof: For some $\pi \in \Pi^0$, we can consider the set of paths $\{\pi' | \pi \prec \pi'\}$ and show by induction on $\prec$ that each path will contain a constraint. It immediately follows that this path has a mating. Since all paths are related in this way to some path in Π^0, the result follows immediately

1. In the first case, all propositional paths are mated – there no constraints.

2. Assume π' occurs at height $n + 1$. By the induction hypothesis, there is a $\pi \prec^n \pi''$ which contains the matched pair of some constraint, and $\pi'' \prec {}^0{}_r\pi'$. There are three cases to consider:

   - If r was not part of the constraint in π'', then the mating for π'' will suffice for π'.
   - If r, was mated in π'', but r does not take part in the constraints of the strong mating for π, then there must be other literals in π'' which do form a constraint, and these still form mating for π'.
   - Otherwise, the new literal, call it r_0, is an essential dominator. If its mate in the associated constraint is in this path, then there is a mating for this path. However, it may have been previously subsumed. There must still be some constraint whose the mated pair is in this set. Assume not. Then some positive r-literal, r_1, in this path dominates the mate for r_0. Another positive r-literal, r_2 dominates the mate for r_1 (there must be one since r_0 is part of a constraint and r_1 dominates it's mate). If we continue this, we eventually get a cycle since there are only a finite number of positive r-literals and, by assumption, no mating. However, this implies the imbedding relationship on the tree is cyclic, and hence there is no MET-proof. This is a contradiction, so some constraint is satisfied in this path, and hence the path is mated.

♣

These results do not take into consideration the possible worlds theory. However, they extend directly. First, we can extend Definition 5.14 by adding the following:

3. For all r dominating l, r is in the x-closure of the associated constraint set for the possible worlds theory x, and all literals upon which the mate for r depends have satisfied constraints.

In other words, if we require transitive closure over some set of literals to get a mated pair, all the literals used in the transitive deduction have satisfied constraints. For reflexivity, our result still holds, since reflexivity adds one literal to a path which, if mated, adds no further constraints. Hence, spanning matings and strong spanning mating are still equivalent. Symmetry is similar.

The result also holds for transitivity, though it is more difficult to see. Transitivity essentially forces paths to triplicate, and adds a positive r-literal to two and a negative r-literal to one of the triplicates. We have to make the allowance that if a transitivity is used, we consider the two alternate paths where the positive r-literals are mated. We put their mates in a set with all other negative r-literals and see that either they have satisfied constraints, or some other literal not depending on transitivity has satisfied constraints. We leave the formal proof for the reader.

By virtue of the results we have obtained, we can now imagine constructing a theorem prover using the following algorithm.

1. Find a mating for Π^0 of an MET

2. For each path in Π^0, attempt to satisfy the constraints for some mated pair in that path.

This means that we do a standard matching process interleaved with some *procedural* code for checking membership in the x-closure of constraint sets. Moreover, there is substantial flexibility in the interaction between searching for matings and strong matings. This method, then, appears to offer more flexibility than standard tableau methods, and also allows construction of MET's

Lastly, this allows use to define a concise version of modal expansion trees. First, we need the following result:

Lemma 5.17 If the restrictions on the accessibility relation are $\exists\forall$, then if $Q := Q_1 \supset Q_2$ is an expansion tree for $\Re_0 \supset \alpha$ and $<_{Q_2}$ is acyclic, then $<_Q$ is acyclic.

This lemma essentially observes that, under certain restrictions, the selection and expansion variables in the "semantic" portion of the tree are not important in the analysis of the imbedding relation.

Proof:

We will prove the contrapositive form of the statement. So, assume that $<_Q$ is cyclic. Then there are x and y in S_Q and Θ_Q such that $x <_Q y$ and $y <_Q x$. Thus, a deduction arc labelled with x dominates a generation arc labelled with y and a deduction arc labelled with y dominates a generation arc labelled with x . But, in the semantic portion of the tree, selection arcs dominate expansion arcs, but not vice-versa. Hence, the cyclic portion of the relation must occur somewhere within the body of the translated formula. Therefore, $<_{Q_2}$ is cyclic. ♣

Thus, we need only consider the MET's imbedding relation. In view of this and the previous results on matings, we can restate the definition of MET proofs as:

Definition 5.16 An MET $\mathcal{M}$ of A is an MET$_x$ *proof* for A if

1. The theory of x if $\exists\forall$
2. $<_{\mathcal{M}}$ is acyclic,
3. The free variable of $\mathcal{M}$ does not appear generated in $\mathcal{M}$.
4. The MET has a strong spanning mating with respect to theory x.

The last condition is the slight strengthening of the notion of tautology we need to make the definition work. In some sense it may seem to be "overspecifying." Standard ET's merely state that the deep formula is tautologous – a truth functional definitional divorced from any procedure. Our condition presupposes a particular procedure for search. However, it seems we need a notion of what p-literals are needed for the proof in order to state a condition like tautology for MET's.

6 Conclusions and Future Research

The thrust of this research has been the formalization of a modal proof theory which is concise yet analytical (or compositional) in nature. The lack (to our knowledge) of other endeavors in this areas has forced our treatment in this thesis has to be necessarily broad. However, it appears that this method is a useful vehicle for generating proofs which yield more intuition of the underlying semantic structure of a formula.

We developed a formal base for theses systems by defining the LKM family. From this we moved to the method of outlines. This yielded a formalism which had a computational nature, and was suitable for implementation of systems with human interaction. Finally, MET's have the promise of providing a "franca lingua" from which proofs in other systems can be generated.

There are, of course, several questions left open by this thesis. Foremost among these is the question of the power, generality and extensibility of such systems. In particular, there what class of languages this approach is sufficient for. For instance, we have investigated a temporal modal language due to Mays [16]. However, at the time of this writing we do not have the soundness proofs of a Gentzen system for this logic. It appears that the a Gentzen system of this form is constructible, and that proof representation and theorem proving along the lines presented this thesis is doable.

The use of outlines and possible automatic generation of Suppes style proofs based on outlines is different approach than is typically seen in automated theorem proving. This style of proof is quite amenable to machine implementation and interaction than standard proof theory. It is arguable that tableau methods are sufficient methods for computer implementation, however a tableau refutation does not (explicitly) return an explanation of the proof. The impetus of the method of outlines is to *demonstrate* the validity of a formula rather than just answer "yes" or "no." This, in turn, could lead to natural language front ends which take a proof and generate an explanation or interact with the user.

There are also the computational issues of designing an outline based proof editor which can accomodate many flavors of modal logic by simple changing of the transform rules and attached semantic procedures. It is possible to envision an MACSYMA or LCF type of system as a tool for logicians based on this method.

The treatment of METs was mostly proof of existance and correctness. There are several questions relating to whether this structure, when extended to the first order case, could be considered a Herbrand type of result for modal logic. We have some initial intuitions along these lines, and it appears that issues of quantification which historically have been difficult to deal with become much clearer in this formalism.

The extended notion of matings presented at the end of chapter 5 seems to offer some interesting new avenues into modal theorem proving. The theory presented there attempts a propositional mating which must then satisfy other "higher order" constraints. However, one can turn the search around and satisfy the relational constraints first. Which way to go is largely a heuristic question – tableau provers being only one point on this spectrum. Moreover, we have started to look at intutionistic logic and its translation to $S4$. It may be possible to construct a method of matings for intuitionistic logic along these lines.

It appears that tableau methods could be used to generate MET's. (We can in fact show that LKM proofs are constructible from tableau refutations). Since tableau methods are the most highly developed computational methods for modal logic, this connection certainly worth investigating.

More than anything, however, the concept of different proof representations and their inter-relationship needs more exploration. In some sense all proof methods are related, though specific methods are tailored toward different purposes. It is of the utmost utility to be able to unify divergent methods and translate between them. This thesis is a beginning toward this unification.

A Implementation Status

A.1 A Natural Deduction Editor

The preceding formalism for generation of natural deduction proofs has been implemented as an interactive outline editor written in prolog. The language is a first order modal logic in which the Barcan formula and it's converse are valid (i.e the domain of all worlds is identical) and all identifiers are rigid. This section presents an overview of the data structures and implementation.

By definition 4.1, a line is a 4-tuple. In prolog, a line is a 5-place term where the fifth member is the world ornamenting the formula. Similarly, a 3-place term containing a list supporting lines, a sponsoring line, and the accessibility relation information. A proof outline is, then, a tuple consisting an (ordered) list of lines, and a list of sequents.

Outline transformations are functions from an outline to another outline. To facilitate definition of these functions, a transform *language* was defined, and a compiler written which translated the transform language to prolog rules. The compiled rules are a relation between the outline, the lines to be transformed, and the new outline resulting from the transformation. The rule fails if some proviso of the outline transform is not satisfied.

Example A.1 The following is an example of a transform rule

```
d_disj iss (  sponsor(H |- C),
              support(H1 |- A ^ B),

              justify(cases),

              add(Line1,[Line1] |- A,stw1,hyp),
              add(Line2,[Line1|H] |- C,sw,NJ),
              add(Line3,[Line3] |- B,stw1,hyp),
              add(Line4,[Line3|H] |- C,sw,NJ1),

              newsequents([(rel;[Line1|gamma]) |- Line2,
                           (rel;[Line3|gamma]) |- Line4])).
```

The reserved words `sw` and `stw1` designate the free world in the sponsoring line, and the first supporting line respectively. The reserved word `rel` designates the relation information in the sequent associated with the sponsoring line.

Proofs can be finished using either the supplied `rulep1` or `rulep2` rules, or `rulep`. In the case of `rulep`, a simple theorem prover is used to verify the validity of the formula.

In order to make such a system usable, a number of addition facilities must be available. This section is a quick and cursory summary of the commands available in the current natural deduction system. This appendix also contains scripts of some example sessions.

A.1.1 Proof Processing Commands

This set of commands can be used to transform old proof outlines into new proof outlines. All transform rules are identified by name. Information required by them (e.g. line numbers, variables) is prompted for automatically. New worlds are of the form wn, and variables are of the form vn where n is some integer. The current transforms with the proof environment are:

rulep	rulep1	rulep2	
pruled	druled		
d_always	p_always		
d_conj	d_disj	d_imp	
d_neg1	d_neg2	d_neg3	d_neg4
d_thinning	d_all	d_exists	

p_conj	p_disj1	p_disj2
p_imp	p_contra	p_neg1
p_neg2	p_neg3	p_neg4
p_ind	p_all	p_exists

Two special rules which need some explanation are `pruled` and `druled`. These are special transforms for introducing definitions into the proof. They prompt for a predicate and generate a new line by replacing (preserving variable bindings) all instances of the predicate by its definition. Definitions are assumed to be in the database in the form:

```
<pred> df <definitions>
```

A.1.2 Informational Commands

The following commands are available for getting information on a proof currently in progress.

- **print** - print the entire proof thus far.
- **active** - print only the active lines in the proof.
- **sequents** - Print the current sequents in the outline.
- **outer** - Print the outer operator of the line which the user supplies to the prompt.

A.1.3 Utility Commands

These rules print information about the outline editor, and allow the user to save, restore, backup, quit, or format a proof.

- **rules** - Print all the deduction rules available.
- **save** - write the proof *lines* to a file which is prompted for. This is different than the `dump` command.
- **dump** - dump the proof to a file which is prompted for. This file can then be the argument of a `restore` command, i.e. it save the entire state of the proof.
- **done** - Quit working on the current proof.
- **tex** - format the proof lines into tex.
- **restore** - This command is typed to the *prolog interpreter* to restore a previously dumped proof. The form is `restore(<fname>)`.

A.1.4 Syntax

The following syntactic conventions are used in the examples.

- and is `&`.
- or is `^`.
- implication is `->>`.
- `~` is not.
- `forall(X,F)` is universal quantification.
- `exists(X,F)` is existential quantification.
- `L` is necessarily.
- `M` is possibly.

A.2 Example Session

A.2.1 A Proof of the S4 Axiom[20]

```
?- prove(l(p) ->> l(l(p))).
rulename: p_imp.
sponsor: _.
rulename: print.
12  12 |- l(p) in w0    hyp
13  12 |- l(l(p)) in w0    _160
11     |- l(p)->>l(l(p)) in w0    deduct(13)
rulename: p_always.
sponsor: _.

world? w1.
```

[20] This is the session used to generate the natural deduction proof of example 4.1.

```
rulename: p_always.
sponsor: _.

world? w2.
rulename: print.
12  12 |- l(p) in w0   hyp
15  12 |- p in w2   _770
14  12 |- l(p) in w1   l-is
13  12 |- l(l(p)) in w0   l-is
11    |- l(p)->>l(l(p)) in w0   deduct(13)
rulename: d_always.
support1: _.

world? w2.
rulename: print.
12  12 |- l(p) in w0   hyp
16  12 |- p in w2   l-ia
15  12 |- p in w2   _770
14  12 |- l(p) in w1   l-is
13  12 |- l(l(p)) in w0   l-is
11    |- l(p->>l(l(p)) in w0   deduct(13)
rulename: tex.
where? 'papers/example.tex'.
rulename: sequents.

r(w1,w2),r(w0,w1); 15 |- [12,16]

rulename: rulep2.
sponsor: _.
support1: _.
The proof is:
```

```
12  12 |- l(p) in w0   hyp
16  12 |- p in w2   l-ia
15  12 |- p in w2   rulep(16)
14  12 |- l(p) in w1   l-is
13  12 |- l(l(p)) in w0   l-is
11     |- l(p)->>l(l(p)) in w0   deduct(13)
```

A.2.2 An Example Using Definitions

```
| ?- assert(a df (b ^ c)).

yes
| ?- prove(a ->> a).
rulename: p_imp.
sponsor: _.
rulename: druled.
support1: _.
Definition?a.
rulename: print.
12  12 |- a in w0   hyp
14  12 |- b^c in w0   def(12)
13  12 |- a in w0   _151
11     |- a->>a in w0   deduct(13)
rulename: pruled.
sponsor: _.
Definition?a.
rulename: print.
12  12 |- a in w0   hyp
14  12 |- b^c in w0   def(12)
16  12 |- b^c in w0   _746
```

```
13  12 |- a in w0   def(16)
11     |- a->>a in w0   deduct(13)
rulename: sequents.
; 14 |- 16
rulename: rulep2.
sponsor: 16.
support1: 14.
The proof is:

12  12 |- a in w0   hyp
14  12 |- b^c in w0   def(12)
16  12 |- b^c in w0   rulep(14)
13  12 |- a in w0   def(16)
11     |- a->>a in w0   deduct(13)
```

A.3 Generating Modal Expansion Trees

The theory discussed in Chapter 5 has been implemented in the form of a Prolog program. This section presents the code for that theorem prover and an example run. This particular implementation is not complete since deduction nodes in trees are limited to a single outgoing arc. Also, the definitions of some predicates such as `member` have been omitted.

```
?- op(20,xfy,r).
?- op(30,fx,~).
?- op(100,xfy,^).
?- op(100,xfy,&).
?- op(150,xfy,->).
?- op(150,xfy,<->).

/* This set of predicates converts to modal NNF */

nnf((~ (A ^ B)), (A1 & A2)) :- nnf((~ A),A1),nnf((~ B),A2).
nnf((~ (A & B)), (A1 ^ A2)) :- nnf((~ A),A1),nnf((~ B),A2).
nnf(~(~ A), B) :- nnf(A,B).
nnf(l(A),l(B)) :- nnf(A,B).
nnf(m(A),m(B)) :- nnf(A,B).
```

```
nnf(~ l(A),m(B)) :- nnf(~A,B).
nnf(~ m(A),l(B)) :- nnf(~A,B).
nnf(~(A -> B) , ( A1 & B1)) :- nnf(A,A1),nnf((~ B),B1).
nnf(~(A <-> B),C) :- nnf((~(A -> B)) ^ (~(B -> A)),C).
nnf((A ^ B), (A1 ^ B1)) :- nnf(A,A1), nnf(B,B1).
nnf((A & B) , (A1 & B1)) :- nnf(A,A1),nnf(B,B1).
nnf((A -> B) , ( A1 ^ B1)) :- nnf(~ A,A1),nnf(B,B1).
nnf((A <-> B),C) :- nnf((A -> B) & (B -> A),C).
nnf(A,A) :- atom(A).
nnf(~A,~A) :- atom(A).

/* An MET is a pair of a tree and a free variable for that tree.  This set
of predicates builds skolemized METs. */

expansion(X,[Y,FV]) :- expansion(X,Y,FV,even,[]),!.

expansion(l(A),gen(Y,L),_,even,DV) :-
                                synskfn(DV,Y),
                                expansion(A,L,Y,even,DV).
expansion(m(A),ded([[X,L]]),_,even,DV) :-
                                expansion(A,L,X,even,[X|DV]).

expansion(A -> B,L1 -> L2,X,even,DV) :-
                                expansion(A,L1,X,odd,DV),
                                expansion(B,L2,X,even,DV).
expansion(A <-> B,L1 <-> L2,X,even,DV) :-
                                expansion(A -> B,L1,X,even,DV),
                                expansion(B -> A,L2,X,even,DV).
expansion(A & B,L1 & L2,X,even,DV) :-
                                expansion(A,L1,X,even,DV),
                                expansion(B,L2,X,even,DV).
expansion(A ^ B,L1 ^ L2,X,even,DV) :-
                                expansion(A,L1,X,even,DV),
                                expansion(B,L2,X,even,DV).
expansion(~ A,~L,X,even,DV)   :- expansion(A,L,X,odd,DV).

expansion(X,Y) :- expansion(X,Y,_,even,[]),!.

expansion(m(A),gen(Y,L),X,odd,DV) :-
                                synskfn(DV,Y),
                                expansion(A,L,Y,odd,DV).

expansion(l(A),ded([[X,L]]),_,odd,DV) :-
                                expansion(A,L,X,odd,[X|DV]).
```

```
expansion(A -> B,L1 -> L2,X,odd,DV) :-
                                expansion(A,L1,X,even,DV),
                                expansion(B,L2,X,odd,DV).
expansion(A <-> B,L1 <-> L2,X,odd,DV) :-
                                expansion(A -> B,L1,X,odd,DV),
                                expansion(B -> A,L2,X,odd,DV).
expansion(A & B,L1 & L2,X,odd,DV) :-
                                expansion(A,L1,X,odd,DV),
                                expansion(B,L2,X,odd,DV).
expansion(A ^ B,L1 ^ L2,X,odd,DV) :-
                                expansion(A,L1,X,odd,DV),
                                expansion(B,L2,X,odd,DV).
expansion(~ A,~L,X,odd,DV)   :- expansion(A,L,X,odd,DV).
expansion(A,A,_,_,_) :- atom(A).

synskfn(Args,Y) :- gensym(sk,A),Y =..[A|Args].

/* An MET for a formula is the skolemized version of the tree for the
   formula in nnf. */

metfor(X,Y) :- nnf(X,Z),expansion(Z,Y).

/* label generates labels for everything in the tree, notes dominance
relationships, and builds an assoc list of literal names and the literals
themselves.  The result is the collapsing of the tree to what corresponds to
the PC transform in this system.  The connection between literals (which
are asserted into various structures in the database) and the atoms is
maintained through the assoc list.  This allows us to use Prolog unification
to do the proper substitution during the theorem proving stage.  Note we
need the occur check.
*/

label([Tree,FV],PCNewTree,PAlist,RAlist) :-
                                FV = sk,
                                label1(Tree,FV,_,PCNewTree,PAlist,RAlist).

label1(A ^ B,FV,Dom,A1 ^ B1,PAlist,RAlist) :-
                                label1(A,FV,Dom,A1,PAlist1,RAlist1),
                                label1(B,FV,Dom,B1,PAlist2,RAlist2),
                                append(RAlist1,RAlist2,RAlist),
                                append(PAlist1,PAlist2,PAlist).

label1(A & B,FV,Dom,C & D,PAlist,RAlist) :-
                               label1(A,FV,Dom,C,PAlist1,RAlist1),
                               label1(B,FV,Dom,D,PAlist2,RAlist2),
                               append(PAlist1,PAlist2,PAlist),
```

```
                              append(RAlist1,RAlist2,RAlist).

label1(gen(NFV,A),FV,Dom,rpair(X,~(FV r NFV)) ^ B,PAlist,
                              [[X,~(FV r NFV)]|RAlist])   :-
                              gensym(rlit,X),
                              gendomrel(X,Dom),
                              label1(A,NFV,Dom,B,PAlist,RAlist).

label1(ded([[NFV,A]|T]),FV,Dom,B ^ C,PAlist,[[X,FV r NFV]|RAlist]) :-
                              T \== [],
                              gensym(rlit,X),
                              assert(dominator(X)),
                              gendomrel(X,Dom),
                              label1(A,NFV,X,B,PAlist1,RAlist1),
                              label1(ded(T),FV,Dom,C,PAlist2,RAlist2),
                              append(PAlist1,PAlist2,PAlist),
                              append(RAlist1,RAlist2,RAlist).

label1(ded([[NFV,A]]),FV,Dom,B,PAlist,[[X,FV r NFV]|RAlist]) :-
                              gensym(rlit,X),
                              assert(dominator(X)),
                              gendomrel(X,Dom),
                              label1(A,NFV,X,B,PAlist,RAlist).

label1(A,FV,Dom,pair(X,B),[[X,B]],[]) :-
                              atom(A),
                                  gensym(plit,X),
                                  B =..[A,FV],
                                  gendomrel(X,Dom).

label1(~ A,FV,Dom,pair(X,~ B),[[X,~ B]],[]) :-
                              atom(A),
                                  gensym(plit,X),
                                  B =..[A,FV],
                                  gendomrel(X,Dom).

/* This asserts facts about the dominance relationship; the reason for the
   var is just  a handy way to handle the case where there is no dominator */

gendomrel(Node,Dominator) :- var(Dominator);
                            nonvar(Dominator),assert(dom(Dominator,Node)).

/*  Do the pi zero paths split into pairs of plits and rlits. */

path(A ^ B,path(R,P)) :-
```

```
                                path(A,path(RA,PA)),
                                path(B,path(RB,PB)),
                                append(RA,RB,R),
                                append(PA,PB,P).

path(A & B,C)  :- path(A,C).
path(A & B,C)  :- path(B,C).

path(pair(L,P),path([],[pair(L,P)])).
path(rpair(L,R),path([rpair(L,R)],[])).

hpaths(PCTree,Paths) :- bagof(Path,path(PCTree,Path),Paths).

/* The structure needed during the proof search is the labelled tree, the
paths, and the dominance relationships */

gen_struct(Tree,Paths,PAlist,RAlist) :-
                        label(Tree,PC,PAlist,RAlist),
                        hpaths(PC,Paths),
                        label_paths(Paths).


/* Ok, some restaints stuff.  In a path, if I mate this literal, what
   else must be mated. */

restraints(Path,Lit,R) :- ppath(Path,path(Rlits,Plits)),
                          dchain(Lit,Rlits,R).

dchain(Lit,Rel,[[Dominator,R1]|R]) :-
                         dom(Dominator,Lit),
                         removedom(Rel,Dominator,R1),
                         dchain(Dominator,R1,R),!.
dchain(_,_,[]).

removedom(Set,Lit,Newset) :- bagof(rpair(Lits,P),
                                   (member(rpair(Lits,P),Set),
                                      \+dom(Lit,Lits)),
                                    Newset).
removedom([],_,[]).

/* Ok, here we go; the theorem proving algorithm. We mate a pair and then
   try to mate the dominators ad infinitum. Note the mutual recursion --
   If you mate some positive r-literal, you have to go satisfy its
   dominating chain. */
```

```
simplemating(Path,Lit1,Lit2) :-
                member(pair(Lit1,Atom),Path),
                member(pair(Lit2,~ Atom),Path).

satisfydominators(Lit,Rlits,RAlist,Mates) :-
                dchain(Lit,Rlits,R),
                satisfy(R,RAlist,Mates).
satisfydominators(tlit(Lit1,Lit2),Rlits,RAlist,Mates) :-
                satisfydominators(Lit1,Rlits,RAlist,Mates),
                satisfydominators(Lit2,Rlits,RAlist,Mates).
satisfydominators(rlit,_,_,_).

satisfy([],_,[]).
satisfy([[Lit,Set]|R],RAlist,[[Lit,Rlit,Other]|Rest]) :-
                        assoc(Lit,RAlist,Atom),
                        in(rpair(Rlit,~ Atom),Set),
                        satisfy(R,RAlist,Rest),
                        satisfydominators(Rlit,Set,RAlist,Other).

/* Defining the closure conditions on the set -- S5 doesn't work without
   more sophistication about marking.  The structure returned contains
   information on what closure rules were used, and what members of the
   set were matched. */

in(X,Set) :- member(X,Set).
in(rpair(reflit,~ X r X),_) :- (t;s4;s5).
in(rpair(tlit(Lit1,Lit2),~ X r Y),Set) :- (s5;s4),
                        member(rpair(Lit1,~ X r Z),Set),
                        in(rpair(Lit2,~ Z r Y),Set).
in(rpair(symlit,~ X r Y),Set) :- s5,in(rpair(_,~ Y r X),Set).

/* So a mating of a propositional path is a simple mating plus satisfaction
   of the extra constraints */

mate(path(Rlits,Plits),mating(Lit1,Mates1,Lit2,Mates2),RAlist) :-
                simplemating(Plits,Lit1,Lit2),
                satisfydominators(Lit1,Rlits,RAlist,Mates1),
                satisfydominators(Lit2,Rlits,RAlist,Mates2).

/* A spanning mating is a mating for all paths. */

spanning_mating([F|R],[F1|R1],RAlist) :-
                mate(F,F1,RAlist),
                spanning_mating(R,R1,RAlist).
spanning_mating([],[],_).
```

```
/* Top level callable predicate.  Given a tree, what is the mating showing it
   tautologous. */

met(X,M) :- cleanup,gen_struct(X,Paths,PAlist,RAlist),
                spanning_mating(Paths,M,RAlist).

prove(X,Z,Y) :- metfor(X,Z),met(Z,Y).

prove(X) :- prove(X,Y,Z),nl,nl,
             write('***************  The Tree ***************'),nl,nl,
            write(Y),nl,nl,
            write('***************   The Proof **************'),nl,nl,
            writemate(Z).

/* *********************   Some utility stuff ************************* */

cleanup :- abolish(dominator,1),abolish(dom,2),abolish(ppath,2).

/* In this case, any literal has only one occurrence, and I don't want it to
   backtrack.  In the general case, the cut has to go. */

assoc(X,List,Y) :- member([X,Y],List),!.

label_paths([F|R]) :- gensym(path,Y),assert(ppath(Y,F)),label_paths(R).
label_paths([]).

/* Hacks to write better */

writemate([]) :- nl.
writemate([mating(X,Y,A,B)|R]) :- write(X),write(' - '),write(A),nl,
                                  write('for '),write(X),write(':'),nl,
                                  writelist(Y),
                                  write('for '),write(A),write(':'),nl,
                                  writelist(B).
```

Example A.2

```
| ?- prove(p -> p).


***************  The Tree ***************
```

```
[~ plit1 ^ plit2,sk]

***************   The Proof **************

plit2 - plit1
for plit2:

for plit1:
```

In this case, there was a propositional mating, and there were no further constraints on the propositions involved.

```
| ?- prove(l(p) -> l(p)).


***************  The Tree ***************

[ded([[sk1,~ plit3]]) ^ gen(sk1,plit4),sk]

***************   The Proof **************

plit4 - plit3
for plit4:

for plit3:
rlit1 rlit2 []
```

In this case, the mating of `plit3` with `plit4` induced the further constraint that two relation literals would be mated – `rlit1` and `rlit2` as indicated. These correspond to the r-literals synthesized from the generation and deduction node in the tree. The empty list indicates that there were no constraints induced by this mating. If there were, they would appear nested within the list.

```
| ?- prove(l(p) -> p).


no
| ?- assert(t).
```

```
yes
| ?- prove(l(p) -> p).


***************  The Tree ***************

[ded([[sk,~ plit7]]) ^ plit8,sk]

***************   The Proof **************

plit8 - plit7
for plit8:

for plit7:
rlit4 reflit []
```

In this case, by changing to system T, we can prove $L(p) \supset p$. In this case mating `plit7` required the use of transitivity as denoted by the `reflit` marker.

```
| ?- prove(l(p) -> l(l(p))).


no
| ?- assert(s4).


yes
| ?- prove(l(p) -> l(l(p))).


***************  The Tree ***************

[ded([[sk5,~ plit11]]) ^ gen(sk4,gen(sk5,plit12)),sk]

***************   The Proof **************

plit12 - plit11
for plit12:
```

```
for plit11:
rlit8 tlit(rlit9,rlit10) []


yes
| ?-
```

Finally, by going into $S4$, we can prove $Lp \supset LLp$. The `tlit` marker indicates transitivity was used, and that it used `rlit9` and `rlit10` from the constraint set.

B Soundness Via First Order Translation

In this section, we present an alternate soundness proof based on the validity preserving properties of a first order translation of the formula. We will define the $\sim$-TRANSFORM for converting a modal propositional statement into the appropriate first-order non-modal expression. This transformation will encode a modal proposition, p, into a first-order predicate $\tilde{p}(w)$, where w signifies the world in which w is true,i.e. $p \overset{\sim}{\Longrightarrow} \tilde{p}(w)$. A modal expression of the form ρp will be transformed into a first order (meta-language) statement expressing the semantics of the (object level) modal statement in terms of the possible-worlds relation R_ρ, i.e. $Lp \overset{\sim}{\Longrightarrow} (\lambda P.\lambda t.\forall x.t\, R_\rho\, x \supset Px)\tilde{p}$.

Definition B.1 The $\sim$-TRANSFORM of a formula, α is:

$$MT(\alpha, 0)w0$$

where MT is defined by:

1. $MT(p, n) := \lambda x_n.\tilde{p}(x_n)$ where p is atomic
2. $MT(\sim p, n) := (\lambda P.\lambda x_n. \sim Px_n)MT(p, n)$
3. $MT(p_1 \oplus p_2, n) := (\lambda P_1 P_2.\lambda x_n.P_1 x_n \oplus P_2 x_n)MT(p_1, n)MT(p_2, n)$ where $\oplus$ is any binary connective.
4. $MT(\rho p, n) := (\lambda P.\lambda x_n.\forall w_n.x_n R_L w_n \supset Pw_n)MT(p, n+1)$ for any universal modality ρ
5. $MT(\omega p, n) := (\lambda P.\lambda x_n.\exists w_n.x_n R_L w_n \wedge Pw_n)MT(p, n+1)$ for any existential modality ω

In the following, we will denote the $\sim$-TRANSFORM of a formula, α, as $\tilde{\alpha}$, and the $\sim$-TRANSFORM of sets of formula, Θ, as $\tilde{\Theta}$.

Example B.1 $LLp \supset Lp \stackrel{\sim}{\Longrightarrow}$
$(\forall x_0.w0R_Lx_0 \supset (\forall x_1.x_0R_Lx_1 \supset \tilde{p}(x_1)) \supset \forall x_0.w0R_Lx_0 \supset \tilde{p}(x_0)$

This approach also extends to first-order modal systems with or without the Barcan formula or its converse. In this case we would simply add a rule which quantifies the bound variable over the $\sim$-TRANSFORM of the matrix. For instance, in the case where the Barcan and it's converse are valid, we would have:

$$MT(\forall y\ \alpha, n) := (\lambda P \lambda x_n \forall y\ P x_n) MT(\alpha, n)$$

We would then have to establish the convention that the predicates are *extended* by one argument which must be a world.

Proposition B.1 Definition B.1 is validity preserving, i.e.

$$\models_{T,S4,S5} \alpha \text{ if } \models \Re_0 \supset \tilde{\alpha}$$

where $\Re$ contains the axiomitization of the appropriate accessibility relation.

Correctness of various forms of semantic translation for satisfiability or validity can be found in [9,4]. In this paper, we will rely on the fact that the $\sim$-TRANSFORM is *validity preserving* .

The basic procedure is to show that an LKM-proof of A can be converted to an LK-proof of $\Re_0 \supset \tilde{A}$.

Proposition B.2 If A has a LKM proof, then $\Re_0 \longrightarrow \tilde{A}$ has an LK proof.

Proof:

1. For a tree of height 0, if $\Re; (p)_w \longrightarrow (p)_w$ is an axiom, then $\Re, MT(p,n)w \longrightarrow MT(p,n)w$ can be converted to an axiom by repeated application of thinning.
2. For a tree of height n, we have the following cases:
   - If the last rule applied was a rule of propositional calculus, the transformation simply of the form:

$$\frac{\Re; \Delta' \longrightarrow \Sigma'}{\Re; \Delta, (\alpha)_w \longrightarrow \Sigma}\ \text{RuleX}$$

or

$$\frac{\Re; \Delta' \longrightarrow \Sigma'}{\Re; \Delta \longrightarrow (\alpha)_w, \Sigma}\ \text{RuleX}$$

These transform to[21]:

$$\frac{\Re, \tilde{\Delta}' \longrightarrow \tilde{\Sigma}'}{\Re, \tilde{\Delta}, MT(\alpha, n) \longrightarrow \tilde{\Sigma}} \quad \text{RuleX}$$

and

$$\frac{\Re, \tilde{\Delta}' \longrightarrow \tilde{\Sigma}'}{\Re, \tilde{\Delta} \longrightarrow MT(\alpha, n), \tilde{\Sigma}} \quad \text{RuleX}$$

The validity of the upper sequents follows from the induction hypothesis.

- For L-IA we have:

$$\frac{\Re; p_{w'}, \Sigma \longrightarrow \Theta}{\Re; (Lp)_w, \Sigma \longrightarrow \Theta} \quad \text{L-IA}$$

transforms to:

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{Given}}{\Re \longrightarrow wR_Lw'}\ \text{RuleX} \qquad \Re, MT(p,n)w', \tilde{\Sigma} \longrightarrow \tilde{\Theta}}{\Re, \Re, wR_Lw' \supset MT(p,n)w', \tilde{\Sigma} \longrightarrow \tilde{\Theta}}\ \forall\text{–IA}}{\Re, wR_Lw' \supset MT(p,n)w', \tilde{\Sigma} \longrightarrow \tilde{\Theta}}\ \text{several thinnings}}{\Re, \forall x.wR_Lx \supset MT(p,n)x, \tilde{\Sigma} \longrightarrow \tilde{\Theta}}\ \forall\text{–IA}$$

By the proviso attached to L-IA, we know that the left branch closes. The upper sequent of the right branch is the ∼-TRANSFORM of the upper sequent of L-IA, and hence closes by the induction hypothesis.

- For L-IS we have:

$$\frac{\Re \cup \{wR_Lw'\}; \Sigma \longrightarrow p'_w, \Theta}{\Re; \Sigma \longrightarrow (Lp)_w, \Theta} \quad \text{L-IS}$$

transforms to:

$$\dfrac{\dfrac{\Re, wR_Lw', \tilde{\Sigma} \longrightarrow \tilde{\Theta}, MT(p,n)w', \tilde{\Gamma}}{\Re, \tilde{\Sigma} \longrightarrow \tilde{\Theta}, wR_Lw' \supset MT(p,n)w', \tilde{\Gamma}}\ \supset\text{–IS}}{\Re, \tilde{\Sigma} \longrightarrow \tilde{\Theta}, \forall x.wR_Lx \supset MT(p,n)x, \tilde{\Gamma}}\ \forall\text{–IS}$$

[21] We have used $\tilde{\Theta}$ to denote $\{MT(\theta,n)w | (\theta)_w \in \Theta\}$.

The proof of the upper sequent follows from the induction hypothesis. The only detail is to ensure that $\forall$–IS is applicable. However, it is easy to show by induction that any free variables in the lower sequent must appear in $\mathfrak{R}$. Hence, by the proviso on L-IS, $\forall$–IS is applicable.

In summary, we have shown that by application of the above transformation rules, we can convert an LKM proof to the corresponding LK proof. ♣

Proposition B.3 If $\vdash_{LKM} \mathcal{A}$, then $\vdash_{LK} \mathfrak{R}_0 \supset \tilde{\mathcal{A}}$.

Proof: Apply Proposition B.2 to the LKM proof. Apply $\supset$ –IS to the bottom sequent in that structure. ♣

Proposition B.4 The LKM systems are sound.

Proof: By Proposition B.3, if we have an LKM proof of α, we have an LK proof of $\tilde{\alpha}$. Proposition B.1 states that Definition B.1 is validity preserving. Hence,

$$\vdash_{LKM} \alpha \Rightarrow \vdash_{LK} \tilde{\mathfrak{R}}_0 \supset \alpha \Rightarrow \models_{t,s4,s5} \alpha$$

♣

C Hilbert Systems for Modal Logic

Here we quickly present sound and complete Hilbert style formulations for the classical and knowledge logics discussed above.

Definition C.1 (A Hilbert System for T, $S4$, and $S5$)

The system T consists of the axioms and inference rules of propositional calculus with the addition of the axioms

$$Lp \supset p$$

$$L(p \supset q) \supset (Lp \supset Lq)$$

and the rule of inference

$$\text{If } \vdash_H p \text{ then } \vdash_H Lp$$

The system $S4$ is T with the addition of the axiom

$$LLp \supset Lp$$

The system $S5$ is $S4$ with the addition of the axiom

$$Mp \supset LMp$$

The following logic of knowledge is a slightly modified version of that appearing in [28].

Definition C.2 (A Hilbert System for Modal Logics of Knowledge)

The basic system consists of the axioms and inference rules of propositional calculus with the addition of the axioms

$$K_i p \supset p$$

$$K_i(p \supset q) \supset (Kip \supset Kiq)$$

and the rule of inference

$$\text{If } \vdash_H p \text{ then } \vdash_H K_i p$$

For common knowledge we add the axiom:

$$Op \supset OK_i p$$

An agent has *positive introspection* if the following axiom appears:

$$K_i p \supset K_i K_i p$$

An agent has *negative introspection* if the following axiom appears:

$$\neg K_i p \supset K_i \neg K_i p$$

References

[1] Andrews, P.B. "Refutations by Matings," *IEEE Transactions on Computers*, C-25, 8 (August 1976), 801-807.

[2] Andrews, P.B. "Theorem Proving via General Matings," *Journal of the Association for Computing Machinery*, 28 (1981), 193-214.

[3] Charniak, E. and McDermott, D. *Introduction to Artificial Intelligence* Reading: Addison-Wesley, 1985.

[4] Chellas, B. F. *Modal Logic, an Introduction* Cambridge: Cambridge University Press, 1980.

[5] Došen K., "Sequent-Systems for Modal Logic," *The Journal of Symbolic Logic*, 50, No. 1, (March 1985).

[6] Gentzen, G. "Investigations into Logical Deductions," *The Collected Papers of Gerhard Gentzen*, ed. M.E. Szabo. North-Holland (1969), 68-131.

[7] Halpern, J.Y., Fagin, R., Vardi, M. Y. "A Model-Theoretic analysis of Knowledge: Preliminary Report, " IBM Research Report RJ373, Yorktown Heights, NY, 1984.

[8] Halpern, J.Y. and Moses, Y.O. "A Guide to the Modal Logics of Knowledge and Belief: Preliminary Draft," *Proceedings of IJCAI 85, Los Angeles.*

[9] Haspel, C.H. *A Study of Some Interpretations of Modal and Intuitionistic Logics in the First Order Predicate Calculus* Syracuse University Ph.D. Dissertation, 1972.

[10] Hintikka, J. *Knowledge and Belief* Ithaca: Cornell University Press, 1962.

[11] Hughes, G.E. and Cresswell, M.J. *An Introduction to Modal Logic.* London: Methuen, 1968.

[12] Hughes, G.E. and Cresswell, M.J. *A Companion to Modal Logic.* London: Methuen, 1984.

[13] Konolige, K. *A Deduction Model of Belief and its Logics.* SRI Technical Note 326, August 1984.

[14] Kripke, S.A. "Semantical Considerations on Modal Logic." *Acta Philosophica Fennica,* 16, 1963.

[15] Mays, E. *A Modal Temporal Logic.* University of Pennsylvania Tech Report MS-CIS-83-29 (1983).

[16] Mays, E., Joshi, A. and Webber, B. *A Modal Temporal Logic for Reasoning About Changing Databases With Application to Natural Langauge Question Answering.* University of Pennsylvania Tech Report MS-CIS-85-01 (1985).

[17] McCarthy, J., Hayes, P.J. "Some Philosophical Problems From the Standpoint of Artificial Intelligence," *Machine Intelligence 4.* Edinburgh University Press, 1969.

[18] Miller, D.A. *Proofs in Higher-Order Logic.* University of Pennsylvania Tech Report MS-CIS-83-37 (1983).

[19] Miller, D.A. *Expansion Tree Proofs and Their Conversion to Natural Deduction Proofs,* University of Pennsylvania Tech Report MS-CIS-84-6 (1983).

[20] Miller, D.A. *Unpublished Classnotes for CIS 682,* University of Pennsylvania, 1985.

[21] Miller, D.A. *Herbrand's Theorem in Higher-Order Logic,* Unpublished draft, July (1985).

[22] Moore, R.C. *Reasoning About Knowledge And Action.* SRI Technical Note 191, October 1980.

[23] Moore, R.C. *A Formal Theory of Knowledge And Action.* SRI Technical Note 320, February 1984.

[24] Morgan, C. G. "Methods for Automated Theorem Proving in Nonclassical Logics," *IEEE Transactions on Computers,* C-25, 8 (August 1976), 852-862.

[25] Ohnishi, M. and Matsumoto, K. "Gentzen Method in Modal Calculi," *Osaka Mathematical Journal,* 9 (1957), 113-130.

[26] Ohnishi, M. and Matsumoto, K. "Gentzen Method in Modal Calculi II," *Osaka Mathematical Journal,* 11 (1959), 115-120.

[27] Prawitz, D. *Natural Deduction.* Uppsala: Alqvist and Wiksells, 1965.

[28] Sato, M. "A study of Kripke-type Models for Some Modal Logics by Gentzen's Sequential Method," *Publications of the Research Institute for Mathematical Science, Kyoto University,* 13 (1977), 381-468.

[29] Zeman, J.J. *Modal Logic* London: Oxford University Press, 1973.