01 Dec 1993

Subsumption in Modal Logic

Dirk Heydtmann

Ralph W. Wilkerson
Missouri University of Science and Technology, ralphw@mst.edu

Subsumption in Modal Logic

D. Heydtmann* and Ralph Wilkerson

CSC-93-33

Department of Computer Science
University of Missouri-Rolla
Rolla, MO 65401

*This report is substantially the M. S. thesis of the first author, completed December 1993

ABSTRACT

Subsumption has long been known as a technique to detect redundant clauses in the search space of automated deduction systems for classical first order logic. In recent years several automated deduction methods for non-classical modal logics have been developed. This thesis explores, how subsumption can be made to work in the context of these modal logic deduction methods.

Many modern modal logic deduction methods follow an indirect approach. They translate the modal sentences into some other target language, and then determine whether there exists a proof in that language, rather than doing deduction in the modal language itself. Consequently, subsumption then needs to focus on the target language, in which the actual proof is done.

World Path Logic (WPL) is introduced as a possible target language. Deduction in WPL works very much like in ordinary logic, the only significant difference is the need for a special purpose unification, which unifies world paths under an equational theory (E-unification). Relating WPL to a well understood first order logic of restricted quantification, the properties of WPL, that make deduction work, are examined. The obtained theoretical results are the basis for the following treatment of subsumption in WPL.

Subsumption is analyzed treating a clause as a scheme standing for the set of its ground instances. Although the notion of ground instances in WPL is different from ordinary logic, it turns out that - just like in ordinary logic - a clause C1 subsumes another clause C2, if there exists a substitution θ such that $C1\theta \subseteq C2$. Once the special purpose unification has been implemented into a theorem prover to allow for deduction in WPL, existing subsumption tests then work without any further changes.

ACKNOWLEDGEMENTS

I want to thank my advisor, Ralph Wilkerson, for his guidance while I was working on this thesis. My thanks go also to the other members of the thesis committee - Bruce McMillin and Michael Hilgers.

Throughout this work I was repeatedly supported by other researchers, who helped me in my literature research, went to great length to make their work available to me, and openly discussed their results with me. These include Marta Cialdea, Luis Fariñas del Cerro, Alan Frisch, Ian Gent, Hans Jürgen Ohlbach, Larry Wos, and most of all, Richard Scherl.

I would also like to thank the American-German Fulbright Kommission in Bonn and the Institute of International Education in Chicago for their financial and visa sponsorship of my graduate studies.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF ABBREVIATIONS

iff	-	if and only if
FOPL	-	First Order Predicate Logic
MGU	-	most general unifier
RML	-	Reified Modal Logic
RML/CL	-	RML Constraint Logic
WLOG	-	without loss of generality
WPL	-	World Path Logic

I. INTRODUCTION

"If you can prove that it's better, it's not worth implementing; and if you can't prove that it's better, it's not worth implementing"

-- *Lincoln A. Wallen*[1]

Even though Leibniz' seventeenth century vision of a symbolic language for the representation and mechanical solution of all scientific and mathematical problems[2] has suffered at the hands of the undecidability and incompleteness[3] results of modern logic, the spirit of his dream lives on within Computer Science. While the decision problem may be theoretically intractable, it has shown practical to prove theorems of symbolic logic mechanically using computers.

Efficiency, however, tends to be a major problem of such *automated theorem provers* or *automated deduction systems*. Their performance depends not only on the fundamental deduction method employed, but can also be improved by various optimization techniques. One of them is called *subsumption*, and is motivated as follows: During the course of a deduction, a large number of sentences is deduced from the given set of premises, until a deduction of the hypothesis is found. At any deduction step, the system has to choose a small subset (usually two) of the available sentences, to perform the next deduction on. Unfortunately, the search space of generated sentences tends to grow rapidly, causing both space and time efficiency problems. The technique of subsumption helps reducing the growth of the search space by detecting redundancies within the set of sentences. The idea is the following: If C can be derived from B and A', where sentence A' is just a variant of a more general or simpler sentence A, then C can also be derived from B and A directly. Thus, A' is not needed and can be discarded. A is said to subsume A'.

---

1) Wallen, personal communication with Ian Gent, 1989, as reported in [Gent 92]

2) for an overview of Leibnizian logic see [Styazhkin 69]

3) [Gödel 31]

Early work in automated deduction focused almost exclusively on classical propositional and first-order logics, and subsumption was treated in this context. In recent years, however, several automated deduction methods for non-classical *modal logics* have been proposed. The relative semantic richness of modal logics makes them suitable for the formalization of a broad variety of human discourses and reasoning, and consequently, modal logics have gained increasing popularity in many areas of computer science and artificial intelligence.

The notion of modal logic can be illustrated very quickly. Basically, modal logic can be viewed as a means to merge language and metalanguage. The concept of modal operators facilitates reasoning about theories, for instance a theory at a certain point of time (in a temporal interpretation), or a theory of an agent's knowledge (in an epistemic interpretation). Conceptually, there are different worlds, each of which has its own truth interpretation, and the modal operators represent an implicit discourse about the accessibility of these worlds.

The goal of this thesis is to explore how the ideas of classical subsumption can be applied to modal logics. Since subsumption as a technique does not make sense per se, but only in the context of a proof system, it cannot be treated independently from the framework of the automated deduction method it is supposed to work in. This is especially important, since the most promising modal logic deduction methods do not perform deduction directly in the modal logic. Instead, they translate the modal language into a special easy to reason about target language and then determine, whether there is a proof in that language. Following this route, the question of subsumption in modal logic deduction reduces to subsumption in the target language.

Our approach is as follows: We present such a target language, called World Path Logic, and demonstrate how modal deduction works in this language. Drawing upon previous results by [Scherl 92], we then show how this World Path Logic can be represented in a first order predicate logic (FOPL) with restricted quantification. FOPL with restricted quantification is relatively well understood, and using the similarity between this language and World Path Logic, we can prove several important properties of World Path Logic.

Building on these results, we finally define subsumption for World Path Logic, show how subsumption can be detected, give an algorithm, and prove the correctness of the method.

It was shown in earlier works that deduction in a language like World Path Logic can be performed very similarly to ordinary first order logic. The only main difference is the need for a special purpose unification routine [Auffray, Enjalbert 89]. It is our contribution to prove that similar results hold for subsumption: As it turns out, Robinson's classical subsumption detection algorithm [Robinson 65] also works for World Path Logic, once the changes to the unification method have been made.

The remaining part of this introduction is devoted to a chapter by chapter outline of the thesis. Chapter II surveys relevant works in the fields of modal logic deduction and of subsumption. While considerable research has been done pertaining to the former, the latter has apparently not received a great deal of attention in the automated deduction community. To our knowledge, this thesis is the first work dealing explicitly with the problem of subsumption in modal logic deduction.

After the general background of modal logics has been presented in Chapter III, modal logic deduction via translation into World Path Logic (WPL) is the subject of Chapter IV. The language of WPL is defined and a translation function from modal logic to WPL is given. The centerpiece of WPL deduction is a special purpose unification method, which unifies terms under an equational theory (E-unification). This method has been adopted from [Auffray, Enjalbert 89]. Chapter IV concludes with a detailed example of a deduction in WPL.

Chapter V presents yet another language, Scherl's RML Constraint Logic (RML/CL) [Scherl 92], as a means for modal logic deduction. RML/CL is less a language for practical applications, but through its well understood theory and closeness to ordinary first order logic it provides valuable insights into how deduction works in languages like WPL. Chapter VI discusses the relationship between WPL and RML/CL further. Drawing on the similarity between WPL and RML/CL, several important properties of WPL deductions are established and proven.

Chapter VII applies the usual definition of subsumption to WPL and establishes a criterion for subsumption detection. Using the theoretical results from Chapter VI, we prove that subsumption in WPL works just like in ordinary first order logic.

Chapter VIII discusses several possible extensions of the method, before finally Chapter IX concludes this thesis with a short summary of the results and some closing remarks.

II. LITERATURE OVERVIEW

This chapter presents a brief overview on relevant works in the areas of (a) automated deduction in modal logics and (b) subsumption. Automated deduction in FOPL has been a well researched field for almost 30 years, since Robinson's landmark paper on resolution [Robinson 65]. Modal logic deduction, however, is a relatively young discipline, with the first considerable work done in 1982 [Fariñas 82]. Since then, a variety of modal deductive methods have been proposed.

These methods can be roughly classified into two groups, the direct and indirect methods [Pelletier 90]. Direct methods establish a proof theory for modal logics, whereas indirect methods translate the modal logic under consideration into some other language - usually a form of FOPL - and then determine whether there is a proof in that target language.

Probably one of the most prominent representative of the direct approach is [Abadi, Manna 86, 90]. The method extends Robinson's resolution method with special inference rules for modal operators. Although Robinson's resolution principle was originally based on formulas in normal form, it can also be stated in terms of non-clausal resolution. Because there is no straight-forward clausal normal form of modal logic, Abadi and Manna's method is based on this non-clausal resolution. An example for one of their modal inference rules is: $(\Box\alpha) \wedge (\Diamond\beta) \vdash \Diamond(\alpha \wedge \beta)$. The intuitive interpretation of this rule is: if α holds in all accessible worlds and β holds in some accessible world, then there must be a reachable world in which both α and β hold. The restrictions on the accessibility relation are represented by the corresponding Hilbert style axioms, for instance $\Box\varphi \vdash \varphi$ in a reflexive system. As [Scherl 92] points out, a major problem of Abadi and Manna's method is the 'cut' rule $\vdash \varphi \vee \neg\varphi$, that is required to make the method complete for first order modal logic. Since the cut rule holds for arbitrary formulas φ, the branching is infinitely large at any point in the search space. Heuristics have to be employed to decide, when and where to apply the cut rule.

Geissler and Konolige [Geissler, Konolige 86] propose a method, where the formula is converted into clausal form as usual, except that this conversion does not effect what lies inside the scope of modal operators. Using special inference rules, the unsatisfiability of a set of sentences S is reduced to unsatisfiability of another set S', such that at least one sentence in S' has less modal operators than in S. The method - or in an implementation the automated theorem prover - is then applied recursively on S', until it has been reduced to an unsatisfiable set of classical logic sentences.

A special characteristic of Geissler and Konolige's approach is the introduction of a so-called bullet operator •. It is attached to variables and skolem terms *within* the scope of a modal operator, if they stem from quantifiers *outside* of the scope of the modal operator. The bullet restricts the way in which unification can be done: if a variable •x is marked, it can only be replaced by a term that is marked itself.

The main drawback of Geissler and Konolige's approach is its recursive nature. An automated theorem prover would need to call itself recursively at each particular resolution step, thus adding considerable complexity. Since a call to a theorem prover is not guaranteed to terminate, it is essential to interleave the calls from one particular step with calls from other steps to maintain completeness [Scherl 92].

Another approach that uses 'semantic' attachment similar to the preceding one, is presented in [Cialdea 86, 91]. Building on earlier work by [Fariñas 82], the method employs a mix of classical and special purpose modal resolution rules. Where Geissler and Konolige use the bullet operator, Cialdea attaches a numerical index to skolem terms and variables, indicating the modal level of the governing quantifier. Again, the index serves as a restriction on unification. A variable can only be unified with a term, if the variable's index is the same or higher than the term's index. As for Geissler and Konolige's method, a binary attachment sufficed, because it would be used anew at every recursive level. Since Cialdea's method does not rely on recursive calls, a numerical attachment is needed.

With respect to the classification into direct and indirect methods, the techniques of Geissler, Konolige, and Cialdea are considered hybrids. They do not translate the modal logic into some other language, but they facilitate the reasoning within the modal language with their special attachments.

Other than the direct and hybrid methods, truly indirect methods translate modal logic into some other representation and then search for a proof in that language, rather than in the modal logic language itself. From their semantics, the modal operators can be interpreted as a quantification over what is usually referred to as 'worlds'. Translation methods make this implicit discourse explicit by translating modal operators into quantifiers. The result is then ordinary first order predicate language or a language very close to it.

The first work using explicit translation for automated theorem proving purposes was reported by [Morgan 76]. The benefit of translation into classical logic is obvious: it makes all the existing deduction machinery available for modal logic. The method of 'naive' translation into FOPL is appealing for its simplicity, but much of the structure and compactness of the original modal formula gets lost over the process of translation. The resulting FOPL expressions are oftentimes very large in size, and inefficient in terms of automated theorem proving.

In recent years indirect modal deduction methods, that translate modal logic into non-classical target logics, have received increasing attention. The target logic gives the worlds special syntactical and semantic consideration, yet it is close enough to FOPL to benefit from existing deduction machinery. Using ideas from [Fitting 83], Jackson and Reichgelt [Jackson, Reichgelt 87, 89] translate the modal operators to indices which are attached to predicates as well as to other terms. Starting from an initial world 0, the respective modal context of a subformula is encoded into a sequence of terms. The $\Box$ operator is replaced by some world variable, say w, and the $\Diamond$ operator by a skolem function of the variables governing it. As an example, $\Box\Diamond P$ is translated into $P^{f(w):w:0}$. Predicates and terms then unify only if their world denoting indices unify.

Auffray and Enjalbert propose a very similar method [Auffray, Enjalbert 89]. What is an index in Jackson and Reichgelt's method, is here stored as an additional argument, called 'path', to predicates and functions. Except for minor syntactical differences (among other things, the order is reversed), this path equals Jackson and Reichgelt's index. Auffray and Enjalbert's important contribution is the concept of E-unification, which describes unifying world paths under an equational theory. The equational theory reflects the specific properties of the world accessibility relation. If this relation is known to be reflexive, for instance, then the equational theory states $w:1 \equiv w$, where 1 is an artificial neutral element. Thus, the paths 0:sk:w and 0:sk E-unify with the substitution $\{1/w\}$. Under the concept of E-unification, each specific accessibility relation calls for its own special purpose unification algorithm.

Ohlbach develops a translation method [Ohlbach 88, 93], in which the accessibility relation is represented in deterministic access functions. Such a function is a one place function that maps worlds into accessible worlds. Since multiple worlds can possibly be accessible from each given world, multiple functions apply to each world. The modal formula $\Box \Diamond P$ is translated into $\forall f\ \exists g\ P(f \circ g)$, such that $f \circ g$ is the composition of the two individual functions, where f returns a world accessible from the initial world and g is a world accessible from that world. Ohlbach's chain of functions corresponds closely to Jackson and Reichgelt's world indices and Auffray and Enjalbert's paths.

A general framework for modal logic deduction has been developed by Frisch and Scherl [Frisch, Scherl 91; Scherl 92]. First order logic with restricted quantification is used in the presence of a restriction theory. Frisch and Scherl do not commit themselves to a particular proof system, they show how a general class of deduction methods for first order logic can be systematically transformed into a modal logic proof system. In particular, they show that the sequence oriented methods of Jackson and Reichgelt, Auffray and Enjalbert, and Ohlbach can be generated as particular instances of the framework. Using insights from constraint logic reasoning, this enables them to show, how sequence unification arises in modal logic.

The work reported in this thesis is based on the sequence oriented methods of Jackson, Reichgelt, and Auffray, Enjalbert. Scherl's work [Scherl 92] provides the theoretical background we utilize for proving certain properties of subsumption in the context of modal logic deduction.

While most of the aforementioned methods are resolution based, Wallen [Wallen 90] proposes a matrix and tableau method, that does not require prenexing, skolemization, and conversion to normal form. This method was later reconstructed and generalized by [Gent 92] based on a logic of restricted quantification similar to the one utilized in [Scherl 92].

In contrast to modal theorem proving, not very many publications deal with subsumption. Robinson's famous paper on resolution [Robinson 65] defines subsumption and gives the subsumption algorithm used in Chapter VII of this thesis. Loveland introduces the notion of θ-subsumption, which is weaker than general subsumption, but more useful for practical purposes [Loveland 78]. He also examines, how subsumption as a deletion strategy effects the underlying resolution strategy within a theorem proving system.

Currently, research is underway as to how temporal subsumption can be used in the context of distributed algorithms verification. The goal is to detect and remove redundant assertions in a verification proof outline [Schollmeyer, McMillin 93]. The temporal model used here, however, is very much tailored to the specific purposes of program fault tolerance.

Summarizing this survey of relevant works in the fields of modal logic deduction and of subsumption, considerable work has been done pertaining to the former, while the latter has apparently not received a great deal of attention in the automated deduction community. To our knowledge, this thesis is the first work dealing explicitly with the problem of subsumption in the context of modal logic deduction.

III. SYNTAX AND SEMANTICS OF MODAL LOGIC

This chapter presents the basic background on modal logic, based upon [Scherl 92, Fariñas 91, Jackson & Reichgelt 89, Wallen 90]. The notion of modal logic has been around for a number of decades, it can be traced back to the works of C.I.Lewis from 1912 to 1932 [Scherl 92]. A pivotal milestone was Kripke's paper on the semantics of modal logic [Kripke 63]. His *possible-worlds* semantics form the basis for nearly all modern modal logic systems. But before we go into semantics, we will first have a brief look at the syntax.

A. Syntax

Modal logic is an extended form of ordinary propositional logic or first-order predicate logic. Throughout this paper, however, we usually mean its first-order version, when we speak of modal logic. The language of modal logic adds two new unary operators, $\Box$ and $\Diamond$, to its FOPL counterpart. These are usually referred to as the operators of *necessity* and of *possibility*, respectively. All well-formed formulas of FOPL are also well-formed formulas in the modal logic language. Additionally, if φ is a well-formed formula, then so are $\Box\varphi$ and $\Diamond\varphi$. For instance, $\exists x \Diamond(P(x) \wedge \Diamond\Box\forall y(Q(y) \vee R))$ is a well formed formula in modal logic.

The operators $\Box$ and $\Diamond$ can be interpreted in multiple ways. If modality is understood to express the concept of *necessity*, then the operators denote two different types of truth. $\Box\varphi$ reads as 'φ is necessarily true', whereas $\Diamond\varphi$ means 'φ is possibly true'. This approach attempts to capture the distinction between things that could not be false (necessary truth), and things, that just happen to be true (contingent truth). In a temporal interpretation $\Box\varphi$ and $\Diamond\varphi$ would be read as 'φ holds *always*' and 'φ will hold *eventually*'. When modal logic is used as a logic of agents and knowledge, then $\Box\varphi$ means 'the agent knows φ'.

All these interpretations have in common the duality between $\Box$ and $\Diamond$, i.e. $\Diamond$ can be expressed in terms of $\Box$: $\Diamond\varphi \equiv \neg\Box\neg\varphi$. So, in essence, the operator $\Diamond$ does not really add semantics to the language. It rather serves as a syntactical convenience.

B. SEMANTICS

The most widely accepted semantics concept for modal logic is Kripke's *possible worlds* semantics [Kripke 63]. Basically, a set of worlds and a binary accessibility relation between worlds is added to the FOPL semantics. Recall that FOPL semantics are given in terms of models. A model M for FOPL is a pair $<D,I>$, such that:

- D is the domain
- I is the interpretation function. If f is an n-ary function[4] in FOPL, then I(f) maps D^n to D. If P is an n-ary predicate in FOPL, then I(P) is a function mapping D^n to {TRUE, FALSE}.

The semantic value of a formula φ under a model $M = <D,I>$, denoted as $[\![\varphi]\!]^M$, is inductively defined as follows:

- $[\![\forall x\varphi]\!]^M$ = TRUE, iff[5] for every $d \in D$: $[\![\varphi\{d/x\}]\!]^M$ = TRUE[6]
  = FALSE, otherwise
- $[\![\exists x\varphi]\!]^M$ = TRUE, iff there exists a $d \in D$: $[\![\varphi\{d/x\}]\!]^M$ = TRUE
  = FALSE, otherwise
- $[\![\neg\varphi]\!]^M$ = TRUE, iff $[\![\varphi]\!]^M$ = FALSE
  = FALSE, otherwise
- $[\![\alpha\rightarrow\beta]\!]^M$ = FALSE, iff $[\![\alpha]\!]^M$ = TRUE and $[\![\beta]\!]^M$ = FALSE
  = TRUE, otherwise
- $[\![P(t_1,\ldots,t_n)]\!]^M$ = $I(P)([\![t_1]\!]^M,\ldots,[\![t_n]\!]^M)$, if P is an n-ary predicate symbol
- $[\![f(t_1,\ldots,t_n)]\!]^M$ = $I(f)([\![t_1]\!]^M,\ldots,[\![t_n]\!]^M)$, if f is an n-ary function symbol
- $[\![t]\!]^M$ is undefined, if t is a variable

In the modal case a model needs to carry information about the worlds and their accessibility. Specifically, a model M for modal logic is a six-tupel $<W,w_0,K,D,D^*,I>$.

---

4) We treat constants as 0-ary functions, and atomic propositions as 0-ary predicates.

5) if and only if

6) $\varphi\{d/x\}$ means: substitute d for every x occurring in φ

- W is the set of worlds
- w_0 is the initial world, a distinguished element of W
- $K \subseteq W^2$ is the accessibility relation. $w_1 K w_2$ iff w_2 is accessible from w_1.
- D is the domain, such that $D = \bigcup_{w \in W} D_w$, where D_w is the domain of world w
- D^* is the domain function, which maps each world $w \in W$ to its domain D_w
- I is a binary interpretation function. Its two arguments are a world w and a term t, which is either an n-ary function or an n-ary predicate. $I_w(t)$ maps D^n to D or to {TRUE, FALSE}, respectively.

In modal logic each world possesses its own domain D_w and its own interpretation I_w. The truth of a formula is always evaluated with respect to a current (or initial) world. Thus, to say a formula is true, is to say, it is true in the initial world of the model under consideration. The other worlds come into play, when the modal operators $\Box$ and $\Diamond$ are used. $\Box\varphi$ is true in the current world, if φ is true in all worlds accessible from the current world. Conversely, $\Diamond\varphi$ is true in the current world, if there is an accessible world in which φ holds.

In order to adapt our FOPL semantics definition for the modal case, we need to replace I by I_{w0}. Furthermore, we need to add the following two lines, which define the meaning of the modal operators $\Box$ and $\Diamond$.

- $[\![\Box\varphi]\!]^M$ = TRUE, iff for every world $w \in W$ such that $w_0 K w$: $[\![\varphi]\!]^{<W,w,K,D,D^*,I>}$ = TRUE
  = FALSE, otherwise
- $[\![\Diamond\varphi]\!]^M$ = $[\![\neg\Box\neg\varphi]\!]^M$

An alternative and more frequently used denotation for $[\![\varphi]\!]^M$ = TRUE is $M \models \varphi$ (read: M *satisfies* φ). A formula φ is *valid* in a model, if it holds in every world of the model, i.e. $\forall_{w \in W} <W,w,K,D,D^*,I> \models \varphi$. A formula is said to be *valid* with respect to a class of models C, if it is satisfied by all models in C. A common classification of models is by the restrictions which are imposed on the accessibility relation. The properties of the accessibility relation are a key factor throughout modal logic, and reflecting this importance, we use them to define different systems of modal logic.

Table I lists some of the possible restrictions along with the axioms, that characterize each of them. If the accessibility does not follow any particular restriction, then we speak of the modal logic K. There is actually one axiom, that holds in all modal logics, thus also in K: *If φ is valid, then so is $\Box\varphi$*. This axiom merely reflects the definition of a formula being valid in a model.

Table I. Accessibility Relation Restrictions and Their Axiom Schemata

Modal Logic	Restriction		Axiom
D	serial	$\forall w_1 \in W \; \exists w_2 \in W \; w_1 K w_2$	$\Box\varphi \rightarrow \Diamond\varphi$
T	reflexive	$\forall w_1 \in W \; w_1 K w_1$	$\Box\varphi \rightarrow \varphi$
B	symmetric	$\forall w_1, w_2 \in W$ if $w_1 K w_2$ then $w_2 K w_1$	$\varphi \rightarrow \Box\Diamond\varphi$
4	transitive	$\forall w_1, w_2, w_3 \in W$ if $w_1 K w_2$ and $w_2 K w_3$ then $w_1 K w_3$	$\Box\varphi \rightarrow \Box\Box\varphi$
5	euclidian	$\forall w_1, w_2, w_3 \in W$ if $w_1 K w_2$ and $w_1 K w_3$ then $w_2 K w_3$	$\Diamond\varphi \rightarrow \Box\Diamond\varphi$

Different restrictions go along with different axioms, as indicated in the table. If we are guaranteed that there is always another world accessible from every world, then we have modal logic D, which is serial, and $\Box\varphi \rightarrow \Diamond\varphi$ is an axiom. This is not trivial, recall that $\Box\varphi$ is vacuously true, if no other world is accessible. Conversely, if we are given the axiom $\Box\varphi \rightarrow \Diamond\varphi$, then it follows that the accessibility relation is serial.

It is not uncommon for a modal logic to abide by several accessibility restrictions. The name of the logic then consists of the individual letters characterizing the restrictions. For instance, the accessibility relation of the logic KTB4 is reflexive, symmetric, and transitive, thus an equivalence relation. The seriality of KTB4 comes basically 'for free', due to the reflexive property.

C. Different Interpretations

1. Temporal Interpretation. Modal logics can be used to formalize a broad variety of human discourses and reasoning. One application is the reasoning about time. It is possible, of course, to deal with time in ordinary FOPL, as in the predicate *it_rains(Seattle, tomorrow)*. The temporal (modal) logic, however, gives the temporal factor special syntactical and semantic consideration.

The modal worlds represent different instances of time, and they are ordered in a linear fashion by the accessibility relation. In this temporal interpretation $\Box\varphi$ means, φ will hold in all futures (always), whereas $\Diamond\varphi$ is interpreted as saying that φ will hold at some future point of time (eventually). $\Box\Diamond$*it_rains(Seattle)* expresses the fact, that at any given future time, it will eventually rain in Seattle, whereas $\Diamond\Box$*it_rains(Seattle)* is a pessimistic view of the big rain, that will come some day and will last forever. Clearly, the accessibility relation is transitive and must be antisymmetric, otherwise we would get caught in "time loops".

Some authors use $\bigcirc$ as a third modal operator. Quoted the 'next' operator, $\bigcirc\varphi$ indicates that φ holds at the very next moment, as opposed to 'eventually' or 'in all futures' [Abadi, Manna 90].

An important application area within computer science, where temporal logics are employed, is the field of program verification, especially with respect to concurrent programs. Most properties about programs, that one would like to prove, fall into two categories [Owicki, Lamport 82]: liveness properties, which state that something good *eventually* does happen, and safety properties, which state that something bad *never* happens. Thus, liveness properties can be expressed in terms of the $\Diamond$ operator, whereas safety properties lend themselves to the $\Box$ operator. Program termination is an example for a liveness property, while mutual exclusion - no two processes are in their critical section at the same time - would be a typical safety property. The question of temporal subsumption in the context of program verification has received some attention lately, the goal here is to increase the efficiency of proof systems for distributed programming [Schollmeyer, McMillin 93].

2. **Epistemic Interpretation and Multimodal Logics.** Another application for modal logics is the reasoning about knowledge of agents. This use of modal logics is usually referred to as an *epistemic* interpretation. In this context, $\Box\varphi$ can be read as: the agent knows φ. Conversely, $\Box\neg\varphi$ means, the agent knows that φ does not hold. This is different from $\neg\Box\varphi$, which states that the agent has no knowledge as to the truth of φ. The other modal operator, $\Diamond$, has no particular interpretation other than a syntactical abbreviation for $\neg\Box\neg$.

While it is nice to have a means to formalize a single agent's knowledge, it is more challenging to deal with multiple agents, each of whom has his own knowledge about the world and about the knowledge of his fellow agents. Representing multiple agents requires distinct modal operators for each agent, which we call $\Box_A$ and $\Box_B$. The formula $\Box_A \varphi$ reads 'agent a knows φ', and $\Box_B \Box_A \varphi$ is read as 'agent b knows that agent a knows that φ holds'.

We will illustrate this with a simple version of the famous Wise-Man Puzzle which is frequently used throughout the literature as a test problem for formalizations of knowledge and belief [Geissler, Konolige 86], [Genesereth, Nilsson 87], [Scherl 92]:

> *There are two wise men who are told by their king that at least one of them has a white spot on his forehead. In fact, both have a white spot. Every wise man can see the other's forehead, but not his own. Suppose wise man B says he does not know whether he has a white spot. The problem is then to prove that A knows he himself has a spot on his forehead.*

The givens are: (i) A knows that if he does not have a spot, B will be aware of that (ii) A knows that B knows that at least one of them has a white spot and (iii) A knows (because B said so) that B does not know whether he has a spot.

These statements can be represented in multimodal logic as follows:

(i) $\Box_A (\neg \mathrm{spot}(A) \rightarrow \Box_B \neg \mathrm{spot}(A))$ (3.1)

(ii) $\Box_A \Box_B (\mathrm{spot}(A) \vee \mathrm{spot}(B))$ (3.2)

(iii) $\Box_A \neg \Box_B \mathrm{spot}(B)$ (3.3)

The hypothesis is:

$$\Box_A \mathrm{spot}(A) \quad (3.4)$$

The treatment in the following chapters is restricted to monomodal logics for simplicity's sake, but in Section VIII.B we will present an extension to multimodal logics and prove the Two Wise-Men Puzzle.

IV. TRANSLATION

A. INTRODUCTION

The possible worlds semantics, as described in Chapter III, treats the modal operators $\Box$ and $\Diamond$ much like a quantification over a set of worlds. Essentially, the operators represent a discourse about worlds and their reachability. The discourse is implicit though, since there are no syntactical entities, like constants or variables, which actually denote the worlds. And indeed, it is definitely not easy to cope with the modal operators in a deduction system without naming the worlds.

A way of making the worlds and the accessibility restrictions syntactically visible is to translate modal formulas into classical FOPL, where the modal operators are converted to explicit quantifications, and the accessibility relation is represented by a new binary predicate, say K( ). $\Box\varphi$ then translates to $\forall w\ (K(0,w) \rightarrow \varphi)$, while $\Diamond\varphi$ is written as $\exists w\ (K(0,w) \wedge \varphi)$, where 0 is the current or initial world. More formally, the translation function $T(\varphi)$ can be recursively defined as shown in table II:

Table II. 'Naive' Translation from Modal Logic into First Order Predicate Logic

$T(\varphi)$	$= t(0,\varphi)$
$t(w,\Box\varphi)$	$= \forall w'\ (K(w,w') \rightarrow t(w',\varphi))$, where w' is an all new variable
$t(w,\Diamond\varphi)$	$= \exists w'\ (K(w,w') \wedge t(w',\varphi))$, where w' is an all new variable
$t(w,\forall x\ \varphi)$	$= \forall x\ t(w,\varphi)$
$t(w,\exists x\ \varphi)$	$= \exists x\ t(w,\varphi)$
$t(w,\alpha \vee \beta)$	$= t(w,\alpha) \vee t(w,\beta)$
$t(w,\alpha \wedge \beta)$	$= t(w,\alpha) \wedge t(w,\beta)$
$t(w,\neg\varphi)$	$= \neg t(w,\varphi)$

As an example, $\Box(\alpha \wedge \Diamond \neg\beta)$ translates to $\forall w_1\ (K(0,w_1) \rightarrow (\alpha \wedge \exists w_2(K(w_1,w_2) \wedge \beta)))$. In addition to translating the formula, we would also need to express the accessibility axioms of the modal logic system under consideration in FOPL. This is straightforward, the transitive logic K4, for instance, needs $\forall w_1,w_2,w_3\ \ K(w_1,w_2) \wedge K(w_2,w_3) \rightarrow K(w_1,w_3)$ to be valid. Once everything has been transformed into FOPL, we can then use all the deduction machinery available for FOPL.

This method of 'naive' translation into FOPL is clearly an indirect modal logic deduction method in terms of the classification used in Chapter II. Direct methods establish a proof theory for modal logics, whereas indirect methods translate the modal logic under consideration into some other language and then determine whether there is a proof in that target language.

'Naive' translation into FOPL is appealing for its simplicity, but much of the structure and compactness of the original modal formula gets lost over the process of translation. The resulting FOPL expressions are frequently very large in size, and inefficient when it comes to automated theorem proving.

In recent years other indirect modal deduction methods that translate modal logic into non-classical target logics, have received increasing attention. The target logic gives the worlds special syntactical and semantic consideration, yet it is close enough to FOPL to benefit from existing deduction machinery. We will be looking at a target logic, in which the worlds are represented by special sequences. If we think of the worlds and the accessibility relation as a digraph, in which the nodes represent the worlds and the edges correspond to the accessibility relation, then a sequence represents the path through the graph from the initial world to the current world.

Our target logic is a slight variation of the *path logic* introduced in [Auffray, Enjalbert 89] and the *sequence representation* used in [Scherl 92]. We will call it *World Path Logic*, abbreviated as WPL. WPL is an efficient language to do modal logic proofs in. Section IV.B introduces WPL and gives a translation procedure from modal logic to WPL. In order to do deduction in WPL, a special unification method is needed. This so-called E-unification is the subject of Section IV.C. Finally, Section IV.D shows by a detailed example, how deduction is done.

The analysis of WPL's key properties, however, is postponed until Chapter VI. The reason is that the theoretical properties of WPL are best being studied by relating it to yet another language, which is introduced in Chapter V. It is Scherl's RML Constraint Logic (RML/CL) [Scherl 92]. RML/CL is less a language for practical applications, but through its well understood theory and closeness to ordinary first order logic it provides valuable insights into how deduction works in languages like WPL.

To keep the presentation simple, we will impose some restrictions on the modal logic under consideration. These restrictions hold from now on until Chapter VIII, in which possible extensions of the method are explored:

- the domain is constant in all worlds
- the logics are monomodal, i.e. there is just one accessibility relation
- the accessibility relation is serial, i.e. from every world there is always another accessible world, and also, we limit the accessibility restrictions to be some combination of reflexivity, symmetry, transitivity. This leaves us with the modal systems KD, KT, KDB, KD4, KTB, KT4, and KT5[7]

B. World Path Logic

Formally, World Path Logic is a three-typed FOPL. The first two types are D, the domain of discourse, and P, the world paths. P is a compound type, the paths are sequences of variables, constants, and functions of type W. W is the set of worlds.

The idea is to attach the access path from the initial world through the world domain to the current world as an additional argument to every predicate and function. As an example, the world path of predicate P in the formula $\Diamond \Box P$ is denoted as: $0 \rightarrow sk \rightarrow w$.

$$\Diamond \Box P \tag{4.1}$$

translates to

$$P(0 \rightarrow sk \rightarrow w) \tag{4.2}$$

0 denotes the initial world, sk is a skolem constant (due to the existential quantification character of the $\Diamond$ operator), and w is a world variable. As a convention throughout this thesis, all variables of type W begin on a 'w', all skolem constants and functions of type W begin on 'sk', and all functions of type D begin on an 'f'.

7) the systems KT4 and KT5 are also referred to as S4 and S5, respectively

Now consider the modal formula

$$\Box \Diamond P \tag{4.3}$$

Naive translation into FOPL yields:

$$\forall w_1 \ (K(0,w_1) \rightarrow \exists w_2 \ (K(w_1,w_2) \wedge P))$$

and after conversion into Skolem[8] Conjunctive Normal Form we obtain:

$$\forall w_1 \ ((\neg K(0,w_1) \vee K(w_1,sk_1(w_1)) \ \wedge \ (\neg K(0,w_1) \vee P)) \tag{4.4}$$

Converting (4.3) into WPL yields

$$P(0 \rightarrow w_1 \rightarrow sk_1) \tag{4.5}$$

which is a much shorter representation of the same semantics as (4.4). The complete translation procedure from modal logic to WPL is given in table III.

As shown in table III, the translation of the $\Diamond$ operator introduces a skolem term like 'sk'. Although this term does not explicitly show any world arguments, the world path preceding sk is considered an implicit argument to sk. Let us look at the expression $\Box \Diamond P$ again. It translates into $P(0 \rightarrow w \rightarrow sk)$. Suppose we wanted to make the implicit argument visible. Then the WPL expression would be $P(0 \rightarrow w \rightarrow sk(0 \rightarrow w))$. But there is no need to write the path $0 \rightarrow w$ twice, because no matter what operation is performed on a world path, the world argument to the skolem function will always be identical to the prefix of the world path leading to and immediately preceding sk. Thus, we just leave the argument out, and consider $0 \rightarrow t_1 \rightarrow \ldots \rightarrow t_n \rightarrow sk(X)$ an abbreviation for $0 \rightarrow t_1 \rightarrow \ldots \rightarrow t_n \rightarrow sk(0 \rightarrow t_1 \rightarrow \ldots \rightarrow t_n, X)$.

Note also, that the procedure distinguishes between rigid and non-rigid predicates and functions. A rigid predicate or function has the same interpretation in all worlds. It does not depend on the current world, and thus, it is not necessary to have the world path as an argument. In all examples given in this thesis, however, we will assume all predicates and functions to be non-rigid, i.e. world dependent.

As an example of the translation procedure, consider the modal expression

$$\Box \Diamond p(x) \ \wedge \ \Diamond \Box (p(g(y)) \rightarrow q(y)) \models \Diamond \forall x \ \Box \Diamond q(x) \tag{4.6}$$

Eventually we want to show, that (4.6) is a theorem in the modal logic KT4. The $\models$ operator is read as 'entails' or 'implies'. The left-hand side of $\models$ is referred to as the set of *premises*, the right-hand side is the *consequent*. More precisely, $\alpha \models \beta$ holds, iff all models, that satisfy α, also

---

8) Skolemization, the technique of eliminiating existential quantifiers, is based on and named after [Skolem 20]

Table III. Conversion Procedure from Modal Logic to World Path Logic

Input: Modal Logic formula φ
Output: World Path Logic formula φ'

1) Close formula, i.e. universally quantify all free variables.
2) convert to Negation Normal Form (move all negation operators inward to the literals)
3) apply translation function $T(\varphi)$ as follows:

• $T(\varphi)$	$= t(0,\varnothing,\varphi)$	
• $t(s,X,\Box\varphi)$	$= t(s\rightarrow w,X,\varphi)$	introduce new variable of type W
• $t(s,X,\Diamond\varphi)$	$= t(s\rightarrow sk(X),X,\varphi)$	Skolemization, sk is an all new function of type W
• $t(s,X,\forall x\ \varphi)$	$= t(s,X\cup\{x\},\varphi)$	add x to the set of universally quantified variables
• $t(s,X,\exists x\ \varphi)$	$= t(s,X,\varphi)\ \{f(s,X)/x\}$	Skolemization, f( ) is an all new function of type D
• $t(s,X,\alpha \vee \beta)$	$= t(s,X,\alpha) \vee t(s,X,\beta)$	
• $t(s,X,\alpha \wedge \beta)$	$= t(s,X,\alpha) \wedge t(s,X,\beta)$	
• $t(s,X,\neg\varphi)$	$= \neg t(s,X,\varphi)$	
• $t(s,X,p(t_1,\ldots,t_n))$	$= p(s,t(s,X,t_1),\ldots,t(s,X,t_n))$	if p is a non-rigid predicate
• $t(s,X,p(t_1,\ldots,t_n))$	$= p(t(s,X,t_1),\ldots,t(s,X,t_n))$	if p is a rigid predicate
• $t(s,X,f(t_1,\ldots,t_n))$	$= f(s,t(s,X,t_1),\ldots,t(s,X,t_n))$	if f is a non-rigid function / constant
• $t(s,X,f(t_1,\ldots,t_n))$	$= f(t(s,X,t_1),\ldots,t(s,X,t_n))$	if f is a rigid function / constant

4) convert formula to clausal form

satisfy β. In order to prepare (4.6) for a later refutation proof, we need to negate the consequent and add it as a conjunct to the premises. This step yields the modal formula:

$$\Box\Diamond p(x)\ \wedge\ \Diamond\Box(p(g(y))\rightarrow q(y))\ \wedge\ \neg\ \Diamond\forall x\ \Box\Diamond q(x) \tag{4.7}$$

As for now, however, we are just concerned about the translation of (4.7) into WPL. Step 1 and 2 convert it into negation normal form. The result is:

$$\begin{aligned} &\Box\Diamond p(x) \\ \wedge\ &\Diamond\Box(p(g(y))\rightarrow q(y)) \\ \wedge\ &\Box\exists x\ \Diamond\Box\ \neg q(x) \end{aligned} \tag{4.8}$$

Application of the translation function T( ) in step 3 of table III yields three clauses:

$$p(0\rightarrow w_1\rightarrow sk_1,x) \tag{4.9}$$

$$\neg p(0 \to sk_2 \to w_2, g(0 \to sk_2 \to w_2, y)) \vee q(0 \to sk_2 \to w_2, y) \tag{4.10}$$

$$\neg q(0 \to w_3 \to sk_3 \to w_4, f(0 \to w_3)) \tag{4.11}$$

This completes the translation into WPL. Next, we will look at unification in WPL.

C. THE CONCEPT OF E-UNIFICATION

Resolution in WPL amounts to classical resolution with a *special purpose unification technique* for world paths. Two WPL predicates unify only if they are possibly in the same world, i.e. if their world paths unify. Consider (4.2) and (4.5). They unify with the *most general unifier* (MGU) $\sigma=\{sk/w_1;sk_1/w\}$. Thus, the common world is sk_1, which is reachable from the initial world 0 through the world sk.

As a specialty of the world path unification method, the elements of two world paths are not always pair-wise unified. The idea is to let the unification method reflect the accessibility axioms.

1. Reflexivity. Consider the two sequences $0 \to sk$ and $0 \to w \to sk$. They do not unify, unless the accessibility relation is guaranteed to be reflexive. In that case, we can safely instantiate w with 0. The resulting path $0 \to 0 \to sk$ is basically equal to $0 \to sk$, because both paths lead to the same world.

More formally, two paths *E-unify*, if they are equal with respect to a certain equational theory.[9] For the case of reflexivity (modal logic KT), a *neutral element* '1' is introduced, and the axiom

$$\forall w \quad w \to 1 \equiv w$$

makes up the equational theory. As for our example, we would instantiate w to the neutral element 1, and obtain the path $0 \to 1 \to sk$. This path is then equal to $0 \to sk$, since we can replace every occurrence of the subsequence $0 \to 1$ by 0 according to the equational theory.

9) The following presentation of E-unification is due to [Auffray, Enjalbert 89]. [Ohlbach 88] deserves credit for the implementational aspects.

An actual implementation of this unification method for reflexivity would have to check for every variable in the path, if deletion of the variable from the path leads to a one-to-one unification, i.e. a pair-wise syntactical identity of the paths under consideration.

It is easy to show that a unique MGU does not always exist. Just consider the paths $0 \rightarrow sk_1 \rightarrow sk_2$ and $0 \rightarrow w_1 \rightarrow w_2 \rightarrow sk_2$. Possible MGUs in this case are $\{1/w_1, sk_2/w_2\}$ and $\{sk_2/w_1, 1/w_2\}$.

2. Symmetry. Now let us examine the symmetric logic KB. Consider the modal formulas $\Diamond \Box \Diamond Q(a)$ and $\neg \Diamond Q(x)$. Their WPL counterparts are $Q(0 \rightarrow sk_1 \rightarrow w_1 \rightarrow sk_2, a(0 \rightarrow sk_1 \rightarrow w_1 \rightarrow sk_2))$ and $\neg Q(0 \rightarrow w_2, x)$. Symmetry tells us that there is a connection from sk_1 back to 0. An instantiation of w_1 to 0 gives us the path $0 \rightarrow sk_1 \rightarrow 0 \rightarrow sk_2$ which is equivalent to $0 \rightarrow sk_2$ and thus unifiable with the second path $0 \rightarrow w_2$, provided sk_2 is substituted for w_2.

Formally, *inverse elements* $(\,)^{-1}$ are introduced, and

$$\forall w,w' \quad w \rightarrow w' \rightarrow w'^{-1} \equiv w$$

is the equality theory for logic KB. Replacing sk_1^{-1} for w_1 in our example yields the predicate $Q(0 \rightarrow sk_1 \rightarrow sk_1^{-1} \rightarrow sk_2, a(0 \rightarrow sk_1 \rightarrow sk_1^{-1} \rightarrow sk_2))$, which can be reduced to $Q(0 \rightarrow sk_2, a(0 \rightarrow sk_2))$ using the equality theory.

Implementationwise, the unification algorithm for symmetric logics has to consider for each variable, if the removal of that variable along with its immediate predecessor leads to a unification.

As with reflexivity, symmetry can lead to multiple MGUs. Consider the paths $0 \rightarrow sk_1 \rightarrow sk_2$ and $0 \rightarrow w_1 \rightarrow w_2 \rightarrow w_3 \rightarrow sk_2$. Possible MGUs are $\{ w_1^{-1}/w_2, sk_1/w_3 \}$ and $\{ w_2^{-1}/w_3, sk_1/w_1 \}$.

3. Path Properties. At this points it becomes clear, why the translation function uses the full path in arguments to (non-world) skolem functions as opposed to just the last world term. Having full paths in arguments to skolem function is what [Auffray, Enjalbert 89] call *strong skolemization*. Consider the predicate $Q(0 \rightarrow sk1 \rightarrow w_1, f(0 \rightarrow sk1 \rightarrow w_1))$ and the substitution $\sigma = \{sk_1^{-1}/w_2\}$. The result is $Q(0,f(0))$. Now apply σ to the same predicate Q without a full path in f( ), i.e. $Q(0 \rightarrow sk1 \rightarrow w_1, f(w_1))$. This time the result is $Q(0,f(sk_1^{-1}))$. The second occurrence of the inverse element could not be resolved with its predecessor, because there was no predecessor present. Therefore, it is important to have the *unique prefix property*:

Theorem 4.1 (Unique Prefix Property) *Multiple occurrences of the same world term always have the same predecessor, and consequently, the same prefix. Also, a variable cannot occur as part of its own prefix. Terms like* $Q(0 \rightarrow w_1 \rightarrow w_2, f(0 \rightarrow w_2 \rightarrow w_1))$ *or like* $0 \rightarrow sk_1 \rightarrow w_2 \rightarrow sk_1$ *cannot occur.*

The *unique prefix property* [Auffray, Enjalbert 89], called *prefix stability* in [Ohlbach 88] follows directly from the translation function. Thus, whenever we substitute an inverse element for a variable, it will resolve with its predecessor at any occurrence of the variable. There will never remain any inverse elements in the path.

Moreover, paths in WPL are inherently linear. They do not branch off as in

$$Q(0 \rightarrow a \rightarrow w,\ f(0 \rightarrow b \rightarrow w')) \tag{4.12}$$

where multiple occurrences of 0 have different successors. Note that (4.12) does not violate the unique prefix property. It cannot occur in WPL though:

Theorem 4.2 (Unique Successor Property) *Within the same WPL literal, all occurrences of a world term t, that have a successor, have the same successor.*

As with the unique prefix property, the unique successor property follows immediately from the translation function. Unification has to preserve both properties.

4. <u>Transitivity</u>. Finally, we consider unification under transitivity (logic K4). It is somewhat more complex than for the cases of reflexivity and symmetry. At first sight, it would appear that this equality axiom will do the job:

$$\forall w,w',w''\quad w \rightarrow w' \rightarrow w'' \equiv w \rightarrow w'' \tag{4.13}$$

Now let us apply this axiom to unify the paths

$$0 \rightarrow a \rightarrow sk \rightarrow b \tag{4.14}$$

and

$$0 \rightarrow w \rightarrow b. \tag{4.15}$$

Substituting sk for w yields the path $0 \rightarrow sk \rightarrow b$, which is equal to the first path under the axiom (4.13). We need to keep in mind, however, that (4.14) is just a short form for

$$0 \rightarrow a \rightarrow sk(0 \rightarrow a) \rightarrow b \tag{4.16}$$

Thus, when sk is substituted for w, it really is $sk(0 \rightarrow a)$, which is inserted into the second path, yielding $0 \rightarrow sk(0 \rightarrow a) \rightarrow b$. But this path violates the unique successor property, because 0 is once succeeded by sk( ), and another time by 'a'.

What needs to be done in order to cope with the transitivity in the previous example, is the substitution of a subsequence of (4.16), $a \rightarrow sk(0 \rightarrow a)$, for w, yielding the unique successor property preserving path

$$0 \rightarrow (a \rightarrow sk(0 \rightarrow a)) \rightarrow b \tag{4.17}$$

(4.17) is equivalent to (4.16), if we make associativity of the binary operator '$\rightarrow$' an axiom:

$$\forall w, w', w'' \quad w \rightarrow w' \rightarrow w'' \equiv w \rightarrow (w' \rightarrow w'') \tag{4.18}$$

Note that the binary infix sequence construct operator '$\rightarrow$' is left-associative. Hence, there are no parentheses needed on the left-hand side of (4.18). An implementation of the unification algorithm for transitivity will mutually try to match up variables of one path to non-empty subsequences of the respective other one. Again, multiple MGUs are possible, but only in a finite number.

5. Combinations of Accessibility Restrictions. As to the combination of any two out of the three properties transitivity, reflexivity, and symmetry, their basic ideas can simply be combined. The equational theory is comprised of the two individual axioms, and the strategies of the implementations are used concurrently.

A special case is the logic KTB4[10], where the accessibility relation has all three properties, i.e. is an equivalence relation. The relation partitions the set of worlds into equivalence classes, out of which we only need to consider the class that contains the initial world 0. The elements in all other classes are unreachable. Within the class that contains 0, the worlds are totally connected. Thus, we can reduce any world path to an equivalent one of length one (0) or two ($0 \rightarrow t$, where t is some world denoting term). The paths $0 \rightarrow w$ and 0 unify with $\{1/w\}$, while the paths $0 \rightarrow t_1$ and $0 \rightarrow t_2$ unify only if the terms t_1 and t_2 unify. Therefore, we have at most one MGU.

---

10) KTB4 is equivalent to KT5, which is more commonly known as S5

D. DEDUCTION IN WPL - AN EXAMPLE

Now that the machinery of world path unification is available, let us go back to proving the theorem (4.6) in the modal logic KT4. Recall:

$$\Box\Diamond p(x) \wedge \Diamond\Box(p(g(y)) \rightarrow q(y)) \models \Diamond\forall x \Box\Diamond q(x) \quad (4.6)$$

We have already done the negation of the right-hand side and translation into WPL, yielding:

$$p(0 \rightarrow w_1 \rightarrow sk_1, x) \quad (4.9)$$

$$\neg p(0 \rightarrow sk_2 \rightarrow w_2, g(0 \rightarrow sk_2 \rightarrow w_2, y)) \vee q(0 \rightarrow sk_2 \rightarrow w_2, y) \quad (4.10)$$

$$\neg q(0 \rightarrow w_3 \rightarrow sk_3 \rightarrow w_4, f(0 \rightarrow w_3)) \quad (4.11)$$

In order to prove (4.6), we can try a refutation resolution proof of the clauses (4.9) through (4.11). Resolution in WPL is basically like ordinary FOPL resolution with the special purpose unification method for world paths. We will start out trying to resolve (4.9) and (4.10). The p()-predicates unify with

$$\sigma = \{ sk_1/w_2,\ sk_2/w_1,\ g(0 \rightarrow sk_2 \rightarrow w_2, y)/x \}$$

Thus, the resolvent of (4.9) and (4.10) is:

$$q(0 \rightarrow sk_2 \rightarrow sk_1, y) \quad (4.19)$$

But the world paths in (4.19) and (4.11) do not unify. So we must start out by E-unifying the q()-predicates in (4.10) and (4.11). Recall that the reflexive and the transitive equality theorem are in the KT4 equation theory. Thus, one of the E-unifiers is:

$$\sigma = \{ sk_3 \rightarrow w_4/w_2\ ,\ sk_2/w_3\ ,\ f(0 \rightarrow w_3)/y \}$$

(4.10) and (4.11) then resolve to:

$$\neg p(0 \rightarrow sk_2 \rightarrow (sk_3 \rightarrow w_4),\ g(0 \rightarrow sk_2 \rightarrow w_2, f(0 \rightarrow w_3))) \quad (4.20)$$

(4.20) and (4.9) unify with

$$\sigma' = \{ sk_2 \rightarrow sk_3/w_1\ ,\ sk_1/w_4\ ,\ g(0 \rightarrow sk_2 \rightarrow w_2, f(0 \rightarrow w_3))/x \}$$

yielding the empty clause ⊠[11] as the resolvent. This concludes the refutation proof of (4.6).

Summing up this chapter, we introduced World Path Logic as a language to perform modal logic deduction in and showed how to translate modal logic formulas into WPL. We presented the concept of E-unification, a special kind of world path unification, which has the

11) we use ⊠ instead of □ to distinguish it clearly from the modal operator □

restrictions on the accessibility relation built into it. Deduction then works very similar to deduction in ordinary FOPL.

World Path Logic as a language is quite similar to languages for modal deduction proposed by other authors [Auffray, Enjalbert 89 and Scherl 92]. Our contributions are:

(a) we use only the last universally quantified world as an argument to skolem functions, not all of them. This point is further elaborated on in theorem 5.1 in Chapter V.

(b) we state the unique successor property (theorem 4.2). As a consequence of this property along with the unique prefix property (theorem 4.1), world paths in world skolem functions equal the prefix of that skolem function in the path, as in $0 \rightarrow w \rightarrow sk(0 \rightarrow w)$. This allows us to simplify the notation: The WPL translation function omits world paths from world skolem functions. $0 \rightarrow w \rightarrow sk$ is then understood as an abbreviation for $0 \rightarrow w \rightarrow sk(0 \rightarrow w)$.

(c) skolemization is integrated into the translation function.

The next chapter will present another language, which will help us to analyze the properties of WPL in Chapter VI.

V. RML CONSTRAINT LOGIC

The previous chapter gave an introduction to World Path Logic. We showed how to translate modal logic into WPL, how deduction works, and presented the special kind of world path unification needed for the deductive process. Doing proofs *in* the language of WPL is one thing, reasoning *about* the language is another. To be able to prove properties of WPL, we first need to develop a deeper insight into its semantics.

One possible way of approaching the semantics is to relate them to modal logic according to the translation function given in table III, and then go from modal logic back to FOPL. Our approach is different though. This chapter will introduce yet another logic, which modal logic can be translated to. It is called *Reified Modal Logic* (RML) [Frisch, Scherl 91]. Basically, it is a constrained form of first order predicate logic. Stressing this fact, we refer to this language as 'RML Constraint Logic' or, in short, RML/CL.

As it turns out, WPL is very close to RML/CL. In fact, there is a direct correspondence between the two languages, and WPL can be viewed as just another syntactical representation of RML/CL. Thus, properties of the relatively easy to reason about constraint first order logic carry over to WPL. This relationship between RML/CL and WPL will be the focus of Chapter VI.

This chapter's presentation of RML/CL is based on [Frisch, Scherl 91] and [Scherl 92]. It is divided into two parts. The first section covers the translation from modal logic into RML/CL, whereas the following section presents *how* to do deduction in RML/CL, and justifies *why* it works.

A. TRANSLATION INTO RML CONSTRAINT LOGIC

As mentioned before, the modal operators □ and ◇ can be seen as an implicit discourse about worlds and their accessibility. In Section IV.A we presented a 'naive' translation into regular FOPL, that made this discourse explicit using quantification over worlds and a special binary predicate K(w1,w2), which can be read as: "world w2 is accessible from world w1". While this translation makes the whole FOPL proof machinery available, the drawback is inefficiency due to far more complex formulas. In addition, it appears that much of the syntactical and semantic structure of modal logic formulas gets lost in the translation.

The underlying idea of *reified modal logic* (RML), a non-modal language with constrained quantifiers, is to capture some of that modal structure by giving the predicate K( ) special syntactical and semantic consideration. There are designated constraint predicates which K() is one of. Actually, K( ) is the only one in a constant domain logic. A varying domain logic, in which a different domain is associated with each world, would require another constraint predicate, e.g. EXIST(x,w), to denote that x is an element of w's domain. To keep things simple, we are only considering constant domains for the time being, thus we do not need the EXIST( ) predicate. Conversion into RML results into a set of constrained sentences plus a constraint theory Σ. Depending on whether a predicate is a constraint predicate or a regular one, it can only occur in designated places. While the constraint predicates, as the name suggests, are only allowed (a) in the constraints and (b) in Σ, the regular predicates can only occur everywhere else.

For instance, $\Box P$ is translated to the constrained sentence $\forall w_{:K(0,w)}\ P(w)$, while $\Diamond P$ translates to $\exists w_{:K(0,w)}\ P(w)$. These sentences are semantically equivalent to $\forall w\ (K(0,w) \rightarrow P(w))$ and $\exists w\ (K(0,w) \wedge P(w))$, respectively. In another example, $\Box \Diamond P$ translates to $\forall x_{:K(0,x)}\ \exists y_{:K(x,y)}$ $P(y)$. This sentence can be read as saying for all x, such that x is a world accessible from the initial world 0, there exists a y, such that y is a world accessible from x, such that P is true in world y.

The translation function into modal logic is given in table IV. Note that every predicate and function has an additional parameter, the current world, to account for changing interpretations in different worlds.

As an example, consider the KT4 modal logic set of sentences from the previous chapter (4.7), which are repeated here as (5.1-5.3).

$$\Box \Diamond p(x) \tag{5.1}$$

$$\Diamond \Box (p(g(y)) \rightarrow q(y)) \tag{5.2}$$

$$\neg \Diamond \forall x \Box \Diamond q(x) \tag{5.3}$$

Translation into RML Constraint Logic yields the sentences:

$$\forall w_{1:K(0,w1)}\ \exists w_{2:K(w1,w2)}\ p(w_2,x) \tag{5.4}$$

$$\exists w_{3:K(0,w3)}\ \forall w_{4:K(w3,w4)}\ p(w_4,g(w_4,y)) \rightarrow q(w_4,y) \tag{5.5}$$

Table IV. Translation Function from Modal Logic to RML Constraint Logic

Input: Modal Logic formula φ
Output: RML Constraint Logic formula $T(\varphi)$

• $T(\varphi)$	$= t(0,\varphi)\ \{0/w_0\}$	substitute 0 for w_0
• $t(i,\Box\varphi)$	$= \forall w_{i+1}{:}K(w_i,w_{i+1})\ \ t(i+1,\varphi)$	
• $t(i,\Diamond\varphi)$	$= \exists w_{i+1}{:}K(w_i,w_{i+1})\ \ t(i+1,\varphi)$	
• $t(i,\forall x\ \varphi)$	$= \forall x\ t(i,\varphi)$	
• $t(i,\exists x\ \varphi)$	$= \exists x\ t(i,\varphi)$	
• $t(i,\alpha\wedge\beta)$	$= t(i,\alpha)\ \wedge\ t(i,\beta)$	
• $t(i,\alpha\ v\ \beta)$	$= t(i,\alpha)\ v\ t(i,\beta)$	
• $t(i,\neg\varphi)$	$= \neg t(i,\varphi)$	
• $t(i,p(t_1,\ldots,t_n))$	$= p(w_i,t(i,t_1),\ldots,t(i,t_n))$	where p is a predicate
• $t(i,f(t_1,\ldots,t_n))$	$= f(w_i,t(i,t_1),\ldots,t(i,t_n))$	where f is a function or constant

$$\neg\ \exists w_{5:K(0,w5)}\ \forall x\ \forall w_{6:K(w5,w6)}\ \exists w_{7:K(w6,w7)}\ \ q(w_7,x) \qquad (5.6)$$

The constraint theory Σ must reflect the restrictions on the accessibility relation for the modal logic system under consideration. Thus, Σ contains one or more of the axioms listed in table V.

Table V. Accessibility Relation Restrictions and Their Axioms in Clausal Form

Modal Logic	Restriction	Axiom	
D	serial	$\forall w_1$	$K(w_1,f(w_1))$
T	reflexive	$\forall w_1$	$K(w_1,w_1)$
B	symmetric	$\forall w_1,w_2$	$K(w_1,w_2) \rightarrow K(w_2,w_1)$
4	transitive	$\forall w_1,w_2,w_3$	$K(w_1,w_2)\ \wedge\ K(w_2,w_3) \rightarrow K(w_1,w_3)$
5	euclidian	$\forall w_1,w_2,w_3$	$K(w_1,w_2)\ \wedge\ K(w_1,w_3) \rightarrow K(w_2,w_3)$

The next step is the conversion of the sentences to *prenex normal form*. While the quantifiers are brought to the outside, the negation operators are moved towards the literals, turning around the quantifiers along the way. This is very much like in ordinary first order predicate logic, with one exception though: Consider the formula

$$(\exists w_{:K(t,w)}\ \alpha)\ \vee\ \beta \tag{5.7}$$

Moving the quantifier outward, i.e.

$$\exists w_{:K(t,w)}\ (\alpha \vee \beta) \tag{5.8}$$

requires that there is actually a world reachable from t. If not, and if β is also true, then the formula (5.7) evaluates to true, while (5.8) is false. If, on the other hand, the seriality axiom is part of the constraint theory Σ, then from every world there is always another world accessible, and moving the quantifiers to the front is safe. A similar argument holds for the formula

$$(\forall w_{:K(t,w)}\ \alpha)\ \wedge\ \beta.$$

The modal logic system of our example, KT4, is not *explicitly* serial, nevertheless the reflexivity axiom guarantees that there will always be an accessible world. Seriality is entailed by reflexivity.

Sentences (5.4) and (5.5) are already in prenex form, and (5.7) converts to

$$\forall w_{5:K(0,w5)}\ \exists x\ \exists w_{6:K(w5,w6)}\ \forall w_{7:K(w6,w7)}\ \neg q(w_7,x) \tag{5.9}$$

Once a formula is in prenex normal form, *skolemization*[12] is used to get rid of the existential quantifiers. We will briefly describe how to eliminate the leftmost existential quantifier; the method can then be used repeatedly to make all of them obsolete. Note that a prenex form sentence is of the form:

$$\forall x_1{:}C_1\ \ldots\ \forall x_{n-1}{:}C_{n-1}\ \exists x_n{:}C_n\ \ \varphi \tag{5.10}$$

where φ is a prenex form formula, $n > 0$, and $\exists x_n$ is the leftmost existential quantification. The C_is are the constraints. They are of the form $K(t,x_i)$, if x_i is a world variable, otherwise the constraint is empty (ordinary quantification).

In the process of skolemization every occurrence of x_n in φ is replaced by a function term $sk(x_1,\ldots,x_{n-1})$, where sk is an all new function symbol, distinct from all other function symbols. The universally quantified variables serve as arguments to the function. However, there is one special consideration that distinguishes skolemization in RML Constraint Logic from non-modal constraint logic: While [Frisch, Scherl 91] use all universally quantified world variables as

12) named after the Norwegian mathematician Thoralf Skolem, who proved that this technique of existential quantifier elimination preserves satisfiability [Skolem 20]

arguments to skolem functions, we will only use the last one. This is justified by the following theorem:

Theorem 5.1 (Only One Relevant World At Each Level) *Let φ be an RML Constraint Logic formula, which has not been skolemized, such that φ is of the form*

$$\forall x_1{:}C_1 \ldots \forall x_{n-1}{:}C_{n-1}\ \exists x_n{:}C_n\ \ \varphi'$$

Let x_i be the highest indexed world variable such that all x_k, where $i<k<n$, are non-world variables. Then the other world variables x_j, where $j<i$, do not occur within the scope of $\exists x_n$, i.e. neither in C_n nor in φ . Furthermore, x_i occurs only in the constraint C_n, not in φ' itself.

Proof: The formula φ is the translation of a modal logic formula. It follows from the translation function listed in table IV, that the constraint C_n consists of $K(x_i,x_n)$ and nothing else. As far as references to worlds in the formula φ', consider for instance the formula $\Box\Box\Diamond\alpha$. Then the interpretation of α depends on the world accessed by the modal operator immediately preceding α. The formula α cannot contain any reference to any other world. In fact, α cannot contain explicit references to worlds at all, because there is no syntactical entity representing the worlds. ■■

Another justification for omitting all but the last world variable in skolem functions is given later on by lemma 6.4. It states that given two access paths from the initial to the current world, where all worlds are ground terms, those paths are equal, if their last world is. In other words, there are no two distinct ground paths ending in the same world. Thus, the other worlds can be viewed as a function of the last world. This standpoint may seem counterintuitive, but lemma 6.4 is restricted to models over the Herbrand Universe. This, however, is all we need to be able to reason about the satisfiability of formulas [Herbrand 30] which is what deduction and theorem proving is all about. Therefore, it suffices to use the last (or current) world in skolem functions. The skolemized version of (5.10) is:

$$\forall x_1{:}C_1 \ldots \forall x_{n-1}{:}C_{n-1}\ (\ \varphi\,\{\ sk(X)/x_n\ \}^{13}\ ) \qquad (5.11)$$

13) This is a substitution. A substitution is generally denoted as φ {t1/x1,...,tn/xn}, where all occurrences of x_i in φ are simultaneously replaced by their respective t_i counterpart

where $X = \{ x_i \mid 1 \leq i \leq n$ and if x_i is a world variable, then there is no other world variable x_j with $i \leq j \leq n \}$.

In addition, information needs to be added to the constraint theory Σ. Recall that $\exists x_{n:Cn}\ \varphi$ is actually an abbreviation for $\exists x_n\ (C_n \wedge \varphi)$. Hence,

$$\forall x_1{:}C_1 \ldots \forall x_{n-1}{:}C_{n-1}\ \ (C_n \wedge \varphi)\ \{sk(\ldots)/x_n\} \tag{5.12}$$

is equal to:

$$(\ \forall x_1{:}C_1 \ldots \forall x_{n-1}{:}C_{n-1}\ \ \varphi\ \{sk(\ldots)/x_n\}\ )\ \wedge\ (\ \forall x_1{:}C_1 \ldots \forall x_{n-1}{:}C_{n-1}\ \ C_n\ \{sk(\ldots)/x_n\}\ ) \tag{5.13}$$

The left conjunct is equivalent to (5.11), whereas the right hand conjunct of (5.13) goes into the constraint theory Σ. Since $\forall x_i{:}C_i\ \alpha$ is just an abbreviation for $\forall x_i\ (C_i \rightarrow \alpha)$, the right hand side of (5.13) can be rewritten as:

$$\forall x_1 \ldots x_{n-1}\ \ C_1 \wedge \ldots \wedge C_{n-1} \rightarrow (\ C_n\ \{sk(\ldots)/x_n\}\ ) \tag{5.14}$$

Each C_i, if not empty, is a K predicate, with the first argument equal to the second argument of the preceding K literal. So the clause added to Σ is of the form:

$$\forall \ldots\ \ K(0,x_1) \wedge \ldots \wedge K(x_{n-2},x_{n-1}) \rightarrow K(x_{n-1},sk(\ldots)) \tag{5.15}$$

In our example, skolemization of the sentences (5.4), (5.5), and (5.9) yields:

$$\forall w_{1:K(0,w1)}\ \ p(sk_1(w_1),x) \tag{5.16}$$

$$\forall w_{4:K(sk2,w4)}\ \ p(w_4,g(w_4,y)) \rightarrow q(w_4,y) \tag{5.17}$$

$$\forall w_{5:K(0,w5)}\ \forall w_{7:K(sk3(w5),w7)}\ \ \neg q(w_7,f(w_5)) \tag{5.18}$$

The above skolemization requires the following sentences to be added to the constraint theory Σ:

$$\forall w_1\ \ K(0,w_1) \rightarrow K(w_1,sk_1(w_1)) \tag{5.19}$$

$$K(0,sk_2) \tag{5.20}$$

$$\forall w_5\ \ K(0,w_5) \rightarrow K(w_5,sk_3(w_5)) \tag{5.21}$$

In addition to these sentences, Σ contains the accessibility axioms from table V for the logic system KT4, i.e. reflexivity and transitivity:

$$\forall w_1\ \ K(w_1,w_1) \tag{5.22}$$

$$\forall w_1,w_2,w_3\ \ K(w_1,w_2) \wedge K(w_2,w_3) \rightarrow K(w_1,w_3) \tag{5.23}$$

Note the following property of the RML constraint theory Σ:

Theorem 5.2 *All of the clauses in the constraint theory Σ are definite clauses.* [Frisch, Scherl 91]

Proof: Clearly, all sentences added by skolemization are of the form of expression (5.15) which is a definite clause. The only other sentences in Σ are those representing the accessibility axioms, as shown in table V. They all are also definite clauses. ■■

The next step after skolemization is the *conversion to clausal form* such that each clause is a disjunction of literals. As usual, we admit the implication operator '→' within clauses, since $\alpha \rightarrow \beta$ is just an abbreviation for $\neg\alpha \vee \beta$. In our actual example however, there are no changes necessary, since the sentences (5.16) to (5.23) are already in clausal form.

Note that a constrained clause of the form

$$\forall w_1{:}K(0,w_1) \ldots \forall w_n{:}K(w_{n-1},w_n) \ \varphi \tag{5.24}$$

is equivalent to

$$\forall w_1 \ldots w_n \ \ (K(0,w_1) \wedge \ldots \wedge K(w_{n-1},w_n)) \rightarrow \varphi \tag{5.25}$$

Now, since it is common to drop universal quantifiers, the above clause can be written as

$$\varphi \ / \ K(0,w_1) \ \wedge \ldots \wedge \ K(w_{n-1},w_n) \tag{5.26}$$

where the right hand side to the slash is the constraint. This convention allows for the elimination of all quantifiers and conversion of the remaining formula into conjunctive normal form. Each clause is associated with a constraint, which is a (possibly empty) conjunction of K predicates.

With this convention the final result of translating our modal logic example into RML Constraint Logic is:

a) the set of constrained clauses:

$$p(sk_1(w_1),x) \ / \ K(0,w_1) \tag{5.27}$$

$$p(w_4,g(w_4,y)) \rightarrow q(w_4,y) \ / \ K(sk_2,w_4) \tag{5.28}$$

$$\neg q(w_7,f(w_5)) \ / \ K(0,w_5) \ \wedge \ K(sk_3(w_5),w_7) \tag{5.29}$$

b) the constraint theory Σ:

$$K(0,w_1) \rightarrow K(w_1,sk_1(w_1)) \tag{5.30}$$

$$K(0,sk_2) \tag{5.31}$$

$$K(0,w_5) \rightarrow K(w_5,sk_3(w_5)) \tag{5.32}$$

$$K(w_1,w_1) \tag{5.33}$$

$$K(w_1,w_2) \ \wedge \ K(w_2,w_3) \rightarrow K(w_1,w_3) \tag{5.34}$$

Summarizing the procedure, the implicit discourse about worlds in a modal logic set of sentences is made explicit by translating the modal operators into constrained quantifications over worlds, with world variables as additional arguments to predicates and functions. The resulting set of constrained sentences is accompanied by a constraint theory Σ, which accommodates the axioms pertaining to the accessibility relation. After converting the set of constrained sentences to prenex form, skolemization eliminates the existential quantifiers. Skolemization of constrained quantified variables requires the addition of clauses to the constraint theory Σ. Finally, the set of constrained sentences is converted to clausal form, and the universal quantifiers are dropped, while their constraints are conjuncted and associated to each clause.

The conversion procedure outlined above *preserves satisfiability*, as was shown by [Frisch, Scherl 91]. This means that a modal logic set of sentences φ is satisfiable if and only if $S \cup \Sigma$ is satisfiable, where S and Σ are the set of constrained sentences and the constraint theory resulting from the conversion into RML Constrained Logic. Based on this translation, the next section will present how deduction works in a constraint logic.

B. Deduction in RML Constraint Logic

This section presents a deduction system for Constraint Logic, that is based on and has been developed from regular FOPL deduction. A central point for the understanding of this section is the relationship between quantified variables and their instances. In a first order logic clause like $P(x) \rightarrow Q(x)$ the implicitly universally quantified variables can be interpreted either as just certain elements in the syntactical structure of the clause, or they can be understood as placeholder such that the clause is viewed as a scheme standing for the set of all its ground instances (a ground instance A of an expression B is a substitution σ into B, such that $A = B\sigma$ is variable-free). This notion is motivated by *Herbrand's Theorem* [Herbrand 30], which states that a set of quantified sentences is satisfiable if and only if the finite set of its ground instances is. Then first order logic deduction can be performed using simple propositional deduction on the set of ground instances. This is rarely done, however. *Unification* is used instead. Ever since the advent of resolution in the 1960s, virtually every automated theorem proving system has used unification to treat universally quantified variables. Where a deduction system for ground instances checks for equality of terms, a corresponding system for quantified sentences tests for unifiability instead. The idea of unification is to instantiate a variable only as far as necessary, delaying the actual choice of ground instances for as long as possible. Thus, deduction on quantified sentences is itself schematic for deduction on ground sentences. As an example consider resolving $P(x,y) \rightarrow Q(x,y)$ with $P(a,f(z))$, yielding $Q(a,f(z))$. Note that every ground instance of the resolvent can also be obtained by resolving two ground instances of the clauses. What is more, every resolvent on the ground level is also an instance of $Q(a,f(z))$.

This relationship between deduction on quantified sentences on the one hand and deduction on ground sentences on the other, is usually formalized in a *lifting lemma*. It states that if S' is a resolvent of S_1' and S_2', and if S_1', S_2' are instances of the quantified sentences S_1 and S_2, then there is a resolvent S of S_1 and S_2 such that S' is an instance of S. In other words, every deduction on the ground instances of a set of sentences can be made schematically from the sentences themselves.

Deduction for Constraint Logic works in a very similar way. The important difference is that a Constraint Logic sentence does not stand for *all* of its ground instances, but only for those that obey the constraints attached to the variables:

Definition 5.1 (Σ-ground Instance) *Let s/C be a constrained sentence, Σ the constraint theory, and σ a substitution such that sσ is ground. Then sσ is a* <u>*Σ-ground instance*</u> *of s/C, iff Cσ is* <u>*Σ-solvable*</u>.

Definition 5.2 (Σ-solvability) *Given a constraint C, C is said to be Σ-solvable, iff there exists a substitution μ such that Cμ is ground and* $\Sigma \models C\mu$.

Definition 5.3 (Set of Σ-ground Instances Σgr') *Σgr'(s/C) is a function mapping s/C to the set of all its Σ-ground instances:* $\Sigma gr'(s/C) = \{s\sigma \mid s\sigma$ *is ground and there exists a μ such that Cσμ is ground and* $\Sigma \models C\sigma\mu\}$.

Notice that Σ is used only to determine whether the constraint is solvable. The Σ-ground instances themselves do not contain any variables nor any constraints.

As to the satisfiability of a set of Constraint Logic sentences, a variant of the Herbrand Theorem applies. The *Constraint Herbrand Theorem* [Frisch, Scherl 91] states that, given a set of constrained skolem normal form sentences S and a constraint theory Σ, $S \cup \Sigma$ is satisfiable if and only if the set of all Σ-ground instances of members of S is satisfiable[14].

Thus, deduction in Constraint Logic could be done performing ordinary propositional resolution on the Σ-ground instances of the constrained sentences. This is valid, because Σ becomes irrelevant once the set of all Σ-ground instances has been obtained.

But then again, we could as well do the deduction schematically on the quantified level. Suppose ordinary FOPL deduction derives $s_3\sigma$ from the sentences s_1 and s_2, where σ is the substitution used in the particular deduction. Then in Constraint Logic, $(s_3 / C_1 \wedge C_2)\sigma$ can be deduced from s_1/C_1 and s_2/C_2, provided the joint constraint $C_1 \wedge C_2$ is Σ-solvable. This is justified by the argument that all resolvents of Σ-ground instances of s_1/C_1 and s_2/C_2 must simultaneously satisfy both constraints. If, however, $C_1 \wedge C_2$ is not Σ-solvable, i.e. no such ground resolvent exists, then $(s_3 / C_1 \wedge C_2)\sigma$ is not a scheme for any derivable ground sentence, and the deduction would not be sound in this case.

What we have described above, is manifested in the *Constraint Lifting Lemma* [Frisch, Scherl 91]: Given a set of constrained clauses S and a constraint theory Σ, if s' is derivable from the Σ-ground instances of S by constraint resolution, then there is a clause s derivable from S, such that s' is a ground instance of s.

Furthermore, constraint resolution is *complete*. If $S \cup \Sigma$ is in fact unsatisfiable, then the empty clause can be derived [Frisch, Scherl 91].

Let us now go back to our example, and try a refutation proof of (4.6), which is repeated here as (5.35)

$$\Box \Diamond p(x) \wedge \Diamond \Box (p(g(y)) \rightarrow q(y)) \vdash \Diamond \forall x \Box \Diamond q(x) \tag{5.35}$$

We have already negated the right hand side, added it to the other two conjuncts on the left, and translated the sentences into skolem normal form RML/CL, yielding S, the set of constrained clauses in (5.27)-(5.29), and the constraint theory Σ in (5.30)-(5.34). The q( ) literals in (5.28) and (5.29) unify with $\sigma = \{ w_4/w_7, f(w_5)/y \}$, yielding the resolvent

$$\neg p(w_4,g(w_4,f(w_5))) \;/\; K(0,w_5) \wedge K(sk_3(w_5),w_4) \wedge K(sk_2,w_4) \tag{5.36}$$

14) Restrictions apply: all constraints must be positive, and Σ must contain definite clauses only. This restriction is met by RML Constraint Logic in most modal systems.

The constraint is Σ-solvable with $\mu = \{ sk_2/w_5, sk_3(sk_2)/w_4 \}$, because

$$\Sigma \models K(0,sk_2) \wedge K(sk_3(sk_2),sk_3(sk_2)) \wedge K(sk_2,sk_3(sk_2))$$

Next, (5.36) is resolved with (5.27). The most general unifier of the p( ) literals is $\sigma = \{ sk_1(w_1) / w_4 , g(sk_1(w_1),f(w_5)) / x \}$, and the resolvent is

$$\boxtimes \;/\; K(0,w_5) \wedge K(sk_3(w_5),sk_1(w_1)) \wedge K(sk_2,sk_1(w_1)) \wedge K(0,w_1) \qquad (5.37)$$

Thus, we have derived the empty clause. We just need to make sure the constraint is Σ-solvable. A possible solution is $\mu' = \{ sk_3(sk_2) / w_1 , sk_2 / w_5 \}$, since

$$\Sigma \models K(0,sk_2) \wedge K(sk_3(sk_2),sk_1(sk_3(sk_2))) \wedge K(sk_2,sk_1(sk_3(sk_2))) \wedge K(0,sk_3(sk_2)). \qquad (5.38)$$

Let us compare this solution with the refutation proof of the same theorem in WPL. There the empty clause was finally derived resolving the two p( ) literals in (4.9) and (4.20) along the world path $0 \rightarrow sk_2 \rightarrow sk_3 \rightarrow sk_1$. As mentioned by the WPL translation function in Chapter IV, the skolem functions have their path prefix as *implicit* world arguments. Thus, $0 \rightarrow sk_2 \rightarrow sk_3 \rightarrow sk_1$ is just an abbreviation for $0 \rightarrow sk_2(0) \rightarrow sk_3(sk_2(0)) \rightarrow sk_1(sk_3(sk_2(0)))$. Notice the resemblance between this path and the terms in the constraint of (5.38). The first conjunct in (5.38) tells us that sk_2 is accessible from 0, the second conjunct suggests that sk_3 is accessed from sk_2, and sk_1 from sk_3. The last two conjuncts follow by the transitive property of logic KT4.

Apparently, (5.38) utilizes the same world path $0 \rightarrow sk_2 \rightarrow sk_3 \rightarrow sk_1$ in some way. There seems to be a close relationship between deduction in WPL and deduction in RML Constraint Logic. This relationship will be the focus of the next chapter.

VI. WORLD PATH LOGIC VS. RML CONSTRAINT LOGIC

The previous chapter presented modal logic deduction via translation into RML Constraint Logic. We showed *how* to do deduction in Constraint Logic, and justified *why* it works. As far as deduction in World Path Logic, Chapter IV covered the '*how*'-part. Explaining *why* it works, is the issue of this chapter.

Our approach is to relate WPL to RML/CL, thus drawing upon the close correspondence between the two languages. We are going to show that a deduction in WPL can be simulated in RML/CL. Thus, Frisch and Scherl's soundness and completeness results for RML/CL, as surveyed in Chapter V, carry over to WPL.[15] The presentation is divided into 4 sections, covering the following topics:

- WPL terms have the same ground instances as corresponding RML/CL terms, where the WPL term's path matches the RML/CL term's constraint (Section A).
- The conjunction of constraints in RML/CL deduction corresponds to the unification of paths in WPL. In particular, we show that (a) a world path resulting directly from the translation represents the same set of worlds as its corresponding constraint in RML/CL, and (b) this identity is preserved over a deduction step. In other words, unification of paths is equivalent to the conjunction of constraints, as far as possible final worlds are concerned (Section B).
- The test for Σ-solvability in RML/CL deduction is replaced by unification in WPL. Two paths unify if and only if the conjunct of the corresponding RML constraints is Σ-solvable (Section C).
- Tieing together the results of the first three sections, we argue that a deduction in WPL can be simulated in RML/CL (Section D).

---

15) It should be noted that the idea of proving properties of one deduction method by relating it to deduction in a first order logic of restricted quantification, is not unique to us. [Gent 92] as well as [Scherl 92] pioneered this approach. In fact, Scherl uses it to prove properties of his world sequence representation. His theorems and proofs are quite different though.

A. Ground Instances

Suppose we translate the same modal logic expressions into WPL and into RML Constraint Logic, using the same skolem function names in both translations. For instance, consider $\varphi = \Box\Diamond q$. While the translation into WPL yields $q(0 \rightarrow w \rightarrow sk_1)$, which is an abbreviation for $q(0 \rightarrow w \rightarrow sk_1(0 \rightarrow w))$, the translation into RML Constraint Logic results in the sentence $S = \{q(sk_1(w)) \mathbin{/} K(0,w)\}$, and in the clause $K(0,w) \rightarrow K(w,sk_1(w))$ as part of the constraint theory Σ. Since this clause is in Σ, we can safely extend the constraint by the conjunct $K(w,sk_1(w))$ without narrowing down the solution space. This gives us the equivalent sentence $q(sk_1(w)) \mathbin{/} K(0,w) \wedge K(w,sk_1(w))$, where the constraint represents the full world access path from the initial world 0 to the world in the q( ) literal, $sk_1(w)$. Note that this path is the same path as in the WPL translation, $q(0 \rightarrow w \rightarrow sk_1(0 \rightarrow w))$. Now, while the translation into WPL does not explicitly set up a constraint theory Σ, we know that for every non-variable term t in the world path, a translation into RML/CL would put a clause with $K(\ldots,t)$ on the right hand side into Σ. This is a key property of WPL, and we refer to it as *Σ-consistency*. A more formal definition of Σ-consistency will follow shortly.

In essence, the information of Σ is stored implicitly in the world paths that result from the translation into WPL. More precisely, it is the part of Σ which is created by skolemization. The remaining part of Σ is the one pertaining to the accessibility restrictions. This information is not stored in the paths, however, it is embedded into the world path unification algorithm. Recall from Section IV.C that the unification algorithm employs special features, depending on the accessibility axioms under consideration. Therefore, Σ is not really needed as an explicit entity in WPL.

Before we define Σ-consistency, we need to go over a few notations regarding paths and accessibility:

Notations: Given a world path P, let P_i denote the i-th term in the path such that $P = P_0 \rightarrow P_1 \rightarrow \ldots \rightarrow P_n$. Then $last(P) = P_n$, and $length(P) = n$. Furthermore, let K be the binary FOPL predicate corresponding to the accessibility relation. The accessibility of P_i from P_{i-1} in terms of FOPL cannot be written as $K(P_{i-1},P_i)$, because the paths would violate the FOPL syntax, but it can be expressed as $K(P'_{i-1},P'_i)$, where P'_j is defined as follows:

- $P'_j = P_j$, if P_j is a variable

- $P'_j = sk_x(Var(P_j))$, if P_j is of the form $sk_x(\ldots)$, where *$Var(P_j)$* is the set of all world variables and non-world variables in the term P_j.

Example: Translate $\varphi = \Box\Diamond\Box\Diamond\, q$ into WPL. This yields q(P) with the path $P = 0 \to w_1 \to sk_1 \to w_2 \to sk_2$, which is an abbreviation for:

$$P = 0 \to w_1 \to sk_1(0 \to w_1) \to w_2 \to sk_2(0 \to w_1 \to sk_1(0 \to w_1) \to w_2).$$

Then $P_4 = sk_2(0 \to w_1 \to sk_1(0 \to w_1) \to w_2)$, and $K(P'_3, P'_4) = K(w_2, sk_2(w_1, w_2))$.

For notational convenience however, we will omit the primes when the meaning is clear. Also, at times we will use K_i as an abbreviation for $K(P'_{i-1}, P'_i)$, when the path is understood. ■■

Definition 6.1 (Σ-consistency) *Given a WPL path R and a constraint theory Σ, R is said to be Σ-consistent, if for any prefix P of R such that length(P) = n and last(P) is a non-variable,*

$$\Sigma \vdash K_1 \wedge \ldots \wedge K_{n-1} \to K_n.$$

A WPL literal is said to be Σ-consistent, if its path is.

As mentioned before, paths resulting from the translation into WPL are Σ-consistent. This is expressed in the following lemma:

Lemma 6.1 (Initial Σ-consistency) *Let P be a prefix of an initial world path, i.e. a world path resulting directly from translation of modal logic into WPL. Then, if last(P) is a non-var,*

$$\Sigma \models K_1 \wedge \ldots \wedge K_{n-1} \to K_n.$$

Proof: Let $P_{v1}, P_{v2}, \ldots, P_{vm}$ be all the variables in P such that v_j is the index of the j-th variable in P. Then

$$K_{v1} \wedge \ldots \wedge K_{vm} \to K_n \tag{6.1}$$

must be a clause in Σ. As an example, consider the path $0 \to w_1 \to sk_1(0 \to w_1) \to w_2 \to sk_2(0 \to w_1 \to sk_1(0 \to w_1) \to w_2)$. The corresponding clauses in Σ are: $K(0,w_1) \to K(sk_1(w_1),w_2)$ and $K(0,w_1) \wedge K(sk_1(w_1),w_2) \to K(w_2,sk_2(w_1,w_2))$. This follows clearly from the RML skolemization procedure.

Now take any literal K_j such that P_j is a non-variable. There must be a clause in Σ with K_j on the right-hand side:

$$K_{v1} \wedge \ldots \wedge K_{vi} \to K_j$$

where $v_i < j$. The left hand side of this clause consists of all the literals K_x, such that $x < j$ and P_x is a variable. Thus, the left hand side is implied by the left-hand side of (6.1), and we can add K_j to the antecedent of (6.1):

$$K_j \wedge K_{v1} \wedge \ldots \wedge K_{vm} \rightarrow K_n$$

After repeated application of this argument we will eventually have extended (6.1) to the clause $K_1 \wedge \ldots \wedge K_{n-1} \rightarrow K_n$. ■■

Up to this point, we have not yet defined the semantics of a WPL expression. For that purpose, we will draw upon the similarity between WPL and Constraint Logic. As previously mentioned, every WPL clause resulting from the translation has a corresponding clause in Constraint Logic. This holds for WPL formulas in general.

Given a WPL literal L(P,...), the corresponding Constraint Logic term is: *LAST*(L) / *constraint*(P) , where *LAST*( ) and *constraint*() are defined as follows:

Definition 6.2 (Function *LAST()*) *Given a WPL literal L, LAST(L) maps L to L' such that L' is the result of the following operation on the syntax of L: (a) copy L to L', (b) replace all world paths P in L' by last(P).*

Note that a WPL literal can contain more than one path, as in the translation of $\Box\exists x \Diamond q$, which is $q(0 \rightarrow w \rightarrow sk_1, f(0 \rightarrow w))$. The other paths are due to skolemization. All those paths, however, are prefixes of the first path. This is guaranteed by the *unique prefix property* (theorem 4.1) and the *unique successor property* (theorem 4.2).

As for this example, remember that every skolem term in a path has all of its prefixes ending in a variable as implicit arguments. $0 \rightarrow w \rightarrow sk_1$ is just an abbreviation for $0 \rightarrow w \rightarrow sk_1(0 \rightarrow w)$. Thus, *LAST*$(q(0 \rightarrow w \rightarrow sk_1, f(0 \rightarrow w))) = q(sk_1(w), f(w))$.

Definition 6.3: (Function *path()*) *The function path() extracts the path out of a WPL literal. Let L be a WPL literal of the form q(P,...). Then path(L) = P.*

Definition 6.4: (Function *constraint()*) The constraint() function converts the path P of a WPL literal into a corresponding conjunction of K literals. Given a path P, constraint(L) = $K_1 \wedge \ldots \wedge K_{length(P)}$.

As an example, let L again be $q(0 \to w \to sk_1, f(0 \to w))$. Then *constraint(path(L))* is $K(0,w) \wedge K(w, sk_1(w))$.

As to the semantics of a WPL literal with variables, we will treat it as a scheme standing for the set of its ground instances. Similar to Constraint Logic, however, we want to consider only instances that are justified by the constraint theory. We will refer to those instances as Σ-ground instances.

More precisely, the set of Σ-ground instances of a WPL literal L should be identical to the set of Σ-ground instances of its Constraint Logic counterpart *LAST(L) / constraint(path(L))*. This is achieved by the following definition:

Definition 6.5 (Σ-ground Instances of WPL, Set Σgr) *Given a WPL literal L and a substitution σ such that Lσ is variable free, LAST(Lσ) is a Σ-ground instance of L if and only if Lσ is Σ-consistent.*

Σgr(L) is a function mapping L to the set of all such Σ-ground instances of L.

Theorem 6.1 (Ground Instance Equivalence) *Given a Σ-consistent WPL literal L, the set of its Σ-ground instances is equal to the set of Σ-ground instances of its Constraint Logic counterpart LAST(L)/constraint(P), i.e.*

$$\Sigma gr(L) = \Sigma gr'(\ LAST(L)/constraint(path(L))\ ).$$

What this theorem says, is that it does not matter whether we go from a path to its ground paths and then convert it into RML Constraint Logic as in *LAST*(Lσ), or if we switch to Constraint Logic first, and then take the Σ-ground instances, i.e. *LAST*(L)σ' such that there exists a grounding substitution μ such that $\Sigma \models$ *constraint(path*(L))$\sigma'\mu$. This is not trivial: if we substitute on the WPL literal directly, then the length of the path can change, as the substitution can contain the neutral element '1', inverse elements, or sub-paths, in the cases of reflexivity, symmetry, and transitivity, respectively. If, however, the substitution is done on the Constraint Logic equivalent, then the length of the access path is predetermined by the number of K-conjuncts in the constraint.

Proof of theorem 6.1:

Part (a): $\Sigma gr(L) \supseteq \Sigma gr'($ *LAST*(L) / *constraint(path*(L)))

Take any element of $\Sigma gr'($ *LAST*(L) / *constraint*(P)) , where P = *path*(L). The element equals *LAST*(L)σ for some substitution σ such that *LAST*(L)σ is ground. Also, there must exist a μ such that *constraint*(P)$\sigma\mu$ is ground. Now perform the same substitutions directly on L, and hence, on the world path P. The joint substitution $\sigma\mu$ cannot contain any special symbols like the neutral element (reflexivity), inverse elements (symmetry), or world path substrings (transitivity), because they are not defined for Constraint Logic. Therefore, *LAST*(L)σ = *LAST*(Lσ) = *LAST*(L$\sigma\mu$), and *constraint*(P)$\sigma\mu$ = *constraint*(P$\sigma\mu$) = *constraint*(*path*(L$\sigma\mu$)). Furthermore, since $\Sigma \models$ *constraint*(P)$\sigma\mu$ by definition of Σ-ground instances in Constraint Logic, $\Sigma \models$ *constraint*(*path*(L$\sigma\mu$)). Thus, *path*(L$\sigma\mu$) is Σ-consistent, and therefore, *LAST*(L$\sigma\mu$) = *LAST*(L)σ is a member of Σgr(L).

As an example, consider $\Box\Diamond$q in modal logic KT. The WPL translation yields L = q(0→w→sk_1), the corresponding constraint theory of translation into Constraint Logic is Σ = {K(w,w); K(0,w)→K(w,sk_1(w)) }. The Constraint Logic equivalent of L is *LAST*(L) / *constraint*(*path*(L)) = q(sk_1(w))/K(0,w) $\wedge$ K(w,sk_1). Its only Σ-ground instance is q(sk_1(0)), where σ = {0/w} and μ = {}. This, however, is also a Σ-ground instance of L, because the path of L$\sigma\mu$, 0→0→sk_1(0), is Σ-consistent. ■

Part (b): $\Sigma gr(L) \subseteq \Sigma gr'($ *LAST*(L)/*constraint*(*path*(L)))

We will prove this by showing that for every grounding substitution σ on a Σ-consistent path Q, where Q=*path*(L), we can construct a grounding substitution σ' for the Constraint Logic counterpart *LAST*(L)/*constraint*(Q) such that:

- **(a)** if Qσ is Σ-consistent then $\Sigma \models$ *constraint*(Q)σ'
- **(b)** *last*(Qσ) = $Q_n\sigma'$, where n=*length*(Q)

The proof is by induction on the cardinality of σ. As the induction hypothesis, suppose the preceding statement holds for all Q, σ as long as $|\sigma| = n$. That is, we can then construct a σ' such that conditions (a) and (b) are met.

The induction hypothesis holds trivially for the base case, where Q is already ground, σ = {}, and σ' = {}.

Now, take another Σ-consistent path P and a singleton substitution μ such that $P\mu = Q$. Then we can construct a μ' such that (a) and (b) hold for P instead of Q, $(\mu\sigma)$ instead of σ, and $(\mu'\sigma')$ instead of σ', thus extending the cardinality of the substitution under consideration to n+1. We need to distinguish four cases:

(i) $\mu = \{1/P_i\}$ and the accessibility relation is reflexive. Then $Q = P\mu = \ldots \rightarrow P_{i-1} \rightarrow P_{i+1} \rightarrow \ldots$ Thus, *constraint*(P) ≡ *constraint*(Q) ∧ $K(P_{i-1},P_i)$. We need to show that

$$\Sigma \models (\ \mathit{constraint}(Q) \wedge K(P_{i-1},P_i)\ )\ \mu'\sigma'$$

for some μ'. Let μ' be $\{\ P_{i-1}/P_i\ \}$. Then, since P_i does not occur in Q, this is equal to

$$\Sigma \models \mathit{constraint}(Q)\ \sigma' \wedge K(P_{i-1},P_{i-1})\ \sigma'$$

The left conjunct holds by the induction hypothesis, the right conjunct follows from the reflexivity axiom in Σ.

For property (b) assume the critical case i=n, i.e. μ substitutes the last term of the world path P. By the induction hypothesis, $\mathit{last}(P\mu\sigma) = P_{n-1}\sigma'$. Conveniently, $P_n\ \mu' = P_{n-1}$. Thus, $\mathit{last}(P(\mu\sigma)) = P_{n-1}\sigma' = (P_n\mu')\sigma' = P_n\ (\mu'\sigma')$. ■

(ii) $\mu = \{P_{i-1}{}^{-1}/P_i\}$ and the accessibility relation is symmetric. Then $Q = P\mu = \ldots \rightarrow P_{i-2} \rightarrow P_{i+1} \rightarrow \ldots$ Thus, *constraint*(Q) ∧ $K(P_{i-2},P_{i-1})$ ∧ $K(P_{i-1},P_i)$ ∧ $K(P_i,P_{i+1})$ ⊨ *constraint*(P). It suffices to show that

$$\Sigma \models (\ \mathit{constraint}(Q) \wedge K(P_{i-2},P_{i-1}) \wedge K(P_{i-1},P_i) \wedge K(P_i,P_{i+1})\ )\ \mu'\sigma'$$

for some μ'. Let μ' be $\{\ P_{i-2}/P_i\ \}$, if P_{i-1} is a non-variable. Otherwise let μ' be $\{P_{i-2}/P_i, f(P_{i-2})/P_{i-1}\}$, where f( ) is the function used in the seriality axiom K(w,f(w)) in Σ. Then, since P_i does not occur in Q, and $K(P_i,P_{i+1})\mu' = K(P_{i-2},P_{i+1})$ is part of *constraint*(Q), this is equal to

$$\Sigma \models \mathit{constraint}(Q)\ \sigma' \wedge K(P_{i-2},P_{i-1})\ \mu'\sigma' \wedge K(P_{i-1},P_{i-2})\ \mu'\sigma'$$

The leftmost conjunct holds by the induction hypothesis, the rightmost literal follows from $K(P_{i-2},P_{i-1})\mu'\sigma'$ by the symmetry axiom in Σ. Now consider the remaining literal $K(P_{i-2},P_{i-1})\mu'\sigma'$. If P_{i-1} is a variable, then $f(P_{i-2})$ is substituted for P_{i-1}, and the literal is entailed by the seriality axiom. Otherwise, if P_{i-1} is not a variable, then $K(P_{i-2},P_{i-1})$ follows from the Σ-consistency of P.

For property (b) assume the critical case i=n, i.e. μ substitutes the last term of the world path P. By the induction hypothesis, $\mathit{last}(P\mu\sigma) = P_{n-2}\sigma'$. Conveniently, $P_n\ \mu' = P_{n-2}$. Thus, $\mathit{last}(P(\mu\sigma)) = P_{n-2}\sigma' = (P_n\mu')\sigma' = P_n\ (\mu'\sigma')$. ■

(iii) $\mu = \{(R_1 \to \ldots \to R_k)/P_i\}$ and the accessibility relation is <u>transitive</u>. Then $Q = P\mu = \ldots \to P_{i-1} \to R_1 \to \ldots \to R_k \to P_{i+1} \to \ldots$ Thus, *constraint*(Q) $\wedge$ $K(P_{i-1},P_i)$ $\wedge$ $K(P_i,P_{i+1})$ $\models$ *constraint*(P). It suffices to show that

$$\Sigma \models (\textit{constraint}(Q) \wedge K(P_{i-1},P_i) \wedge K(P_i,P_{i+1}))\ \mu'\sigma'$$

for some μ'. Let μ' be $\{ R_k/P_i \}$. Then, since P_i does not occur in Q, this is equal to

$$\Sigma \models \textit{constraint}(Q)\ \sigma' \wedge K(P_{i-1},R_k)\sigma' \wedge K(R_k,P_{i+1})\sigma'$$

The leftmost conjunct holds by the induction hypothesis, the rightmost literal is already part of *constraint*(Q). $K(P_{i-1},R_k)$ follows from *constraint*(Q) by the transitivity axiom in Σ, since $K(P_{i-1},R_1) \wedge \ldots \wedge K(R_{k-1},R_k)$ is part of *constraint*(Q).

For property (b) assume the critical case i=n, i.e. μ substitutes the last term of the world path P. By the induction hypothesis, *last*$(P\mu\sigma) = R_k\sigma'$. Conveniently, $P_n\ \mu' = R_k$. Thus, *last*$(P(\mu\sigma)) = R_k\sigma' = (P_n\mu')\sigma' = P_n\ (\mu'\sigma')$. ■

(iv) $\mu = \{t/P_i\}$, such that non of the cases (i) - (iii) applies. In other words, t is an ordinary world term. In that case let $\mu' = \mu$, and the properties (a) and (b) follow trivially. This completes the proof of theorem 6.1. ■■

B. <u>Path Unification vs. Conjunction of Constraints</u>

Both WPL and RML/CL use world terms to make modal logic's implicit discourse about possible worlds visible. While RML/CL restricts world terms by explicit constraints and a separate constraint theory Σ, the world paths serve a similar purpose in WPL. When it comes to deduction, WPL uses the unification of world paths, whereas a resolution step in RML/CL requires the conjunction of two constraints, plus the unification of their last world term. In effect, both methods have a deduction step narrow down the set of possible worlds.

This section's goal is to show two things: (a) a world path resulting directly from the translation represents the same set of worlds as its corresponding constraint in RML/CL, and (b) this identity is preserved over a deduction step. In other words, unification of paths is equivalent to the conjunction of constraints as far as possible final worlds are concerned.

As an example, consider the translation of $\Diamond \Box q \wedge \Box \Diamond \neg q$ which results in the set of RML/CL sentences S =

$$q(w_1) \;/\; K(sk_1,w_1) \tag{6.2}$$

$$\neg q(sk_2(w_2)) \;/\; K(0,w_2) \tag{6.3}$$

and the constraint theory Σ =

$$K(w,f(w)) \tag{6.4}$$

$$K(0,sk_1) \tag{6.5}$$

$$K(0,w) \rightarrow K(w,sk_2(w)) \tag{6.6}$$

Given this Σ, S can be rewritten in this equivalent extended form:

$$q(w_1) \;/\; K(0,sk_1) \wedge K(sk_1,w_1) \tag{6.7}$$

$$\neg q(sk_2(w_2)) \;/\; K(0,w_2) \wedge K(w_2,sk_2(w_2)) \tag{6.8}$$

where each constraint corresponds to the full access path from the initial world 0 to the respective current world. Resolution of (6.7) and (6.8) with the unifier $\sigma = \{ sk_2(w_2)/w_1 \}$ yields:

$$\boxtimes \;/\; K(0,sk_1) \;\wedge\; K(sk_1,sk_2(w_2)) \;\wedge\; K(0,w_2) \;\wedge\; K(w_2,sk_2(w_2)) \tag{6.9}$$

The next thing to do is to check Σ-solvability of the constraint in (6.9). This means finding a grounding substitution for the constraint such that the constraint is entailed by Σ. Note that the first literal, $K(0,sk_1)$, is an instance of the third, $K(0,w_2)$. If the first literal is true, then we do not need to worry about the third. We just make sure the substitution contains $\mu = \{sk_1/w_2\}$, thus making the third and first literal equal. Conveniently, μ also unifies the other two literals. Hence, the problem is reduced to testing the Σ-solvability of

$$\Sigma \models K(0,sk_1) \;\wedge\; K(sk_1,sk_2(sk_1)) \tag{6.10}$$

This constraint is already ground, and it also follows from Σ. Thus, (6.9) is Σ-solvable, and the empty clause $\boxtimes$ has been successfully deduced.

Now translate the same modal formula, $\Diamond \Box q \;\wedge\; \Box \Diamond \neg q$, into WPL:

$$q(0 \rightarrow sk_1 \rightarrow w_1) \tag{6.11}$$

$$\neg q(0 \rightarrow w_2 \rightarrow sk_2) \tag{6.12}$$

These two clauses unify along the path $0 \rightarrow sk_1 \rightarrow sk_2$, resolving into the empty clause. Note the similarity between this ground path and the grounded constraint in (6.10). We want to prove that both, the unified paths of (6.11) and (6.12) as well as the conjuncted constraints in (6.10), necessarily end in the same world, sk_2.[16] But first, we need to define what is meant by the worlds that a path or a constraint can end in.

16) actually in $sk_2(sk_1)$. Recall that $0 \rightarrow sk_1 \rightarrow sk_2$ is an abbreviation for $0 \rightarrow sk_1 \rightarrow sk_2(sk_1)$.

Definition 6.6 (Ground Last Worlds of a Path) *Given a path P, GrLW(P) denotes the set of possible ground last worlds of that path:*

$$GrLW(P) = \{ last(P\sigma) \mid P\sigma \text{ is } \Sigma\text{-consistent and ground} \}$$

Given a set of paths S, $GrLW(S) = \bigcup_{P \in S} GrLW(P)$.

Example: Consider the paths in (6.11) and (6.12), and assume a serial logic KD. Then $GrLW(0 \rightarrow sk_1 \rightarrow w_1) = \{ sk_2(sk_1), f(sk_1) \}$[17] and $GrLW(0 \rightarrow w_2 \rightarrow sk_2) = \{ sk_2(sk_1), sk_2(f(0)) \}$.

Definition 6.7 (Ground Last Worlds of a Constraint) *Given a world term t and a constraint C constraining t, (t occurs in the rightmost K literal of the constraint), GrLW(t,C) denotes the set of all possible ground instances of t such that the constraint C is Σ-solvable:*

$$GrLW(t,C) = \{t\sigma \mid t\sigma \text{ is ground and } \exists\mu \text{ such that } \Sigma \vdash C\sigma\mu\}$$

As an example, consider the constraints from (6.7) and (6.8). $GrLW(w_1, K(0,sk_1) \wedge K(sk_1,w_1)) = \{ sk_2(sk_1), f(sk_1) \}$, and $GrLW(sk_2(w_2), K(0,w_2) \wedge K(w_2,sk_2(w_2))) = \{ sk_2(sk_1), sk_2(f(0)) \}$.

Notice that the examples to definition 6.7 result in the same world sets as the examples to definition 6.6. The next theorem states this identity, i.e. it justifies that the set of possible last worlds of a path P is equal to the set of possible last worlds of P's RML/CL counterpart *constraint*(P):

Theorem 6.2 *Given a WPL path P and a constraint theory Σ, GrLW(P) = GrLW(P_n,constraint(P)).*

Proof: Take an arbitrary predicate, say '*dummy( )*', and construct the WPL literal *dummy*(P). Its Σ-ground instances are equal to GrLW(P). More precisely: *dummy*(t) ∈ Σgr(*dummy*(P)) if and only if t ∈ GrLW(P). This follows by the definition of Σgr( ) and GrLW().

Next, construct the corresponding RML/CL predicate *dummy*(P_n)/*constraint*(P). Then its Σ-ground instances are also equal to GrLW(P_n,*constraint*(P)): *dummy*(t) ∈

17) f( ) is assumed to be the function used in the seriality axiom K(w,f(w))

$\Sigma gr'(dummy(P_n)/constraint(P))$ if and only if $t \in GrLW(P_n,P)$. This again follows from the definitions of $\Sigma gr'()$ and $GrLW()$.

Now it suffices to show that $\Sigma gr(dummy(P)) = \Sigma gr'(dummy(P_n) / constraint(P))$. But this is guaranteed by theorem 6.1. ■■

With the proof of the above theorem we have reached the first goal of this section. Based on the findings of Section A we know that the translation of modal logic into WPL has a path P, where the translation into RML/CL has a constraint equivalent to *constraint(P)*. And theorem 6.2 states that both represent the same set of final worlds.

The next theorem which we are working towards will establish what we intended to prove as the second goal of this section: Unification of paths is equivalent to the conjunction of constraints, as far as possible final worlds are concerned. This proof uses the lemmas 6.2 to 6.4, which will be presented next.

Consider the world $sk_2(sk_1))$, which is the solution of (6.9), where two constraints are joined. Note that it is identical to the intersection of the two sets pertaining to the two individual constraints, as listed in the example to definition 6.7. As one might expect, this is not a coincidence. The following lemma establishes this relationship:

Lemma 6.2 (Ground Worlds Set Intersection) *Given two constrained world terms, t_1 and t_2, along with their constraints, C_1 and C_2, then:*

$$GrLW(t_1\mu,\ C_1\mu \wedge C_2\mu) = GrLW(t_1,C_1) \cap GrLW(t_2,C_2)$$

where μ is the MGU of t_1 and t_2.

Proof: ($\subseteq$) Take any w such that $w \in GrLW(t_1\mu,\ C_1\mu \wedge C_2\mu)$. Thus, $w = t_1\mu\alpha$ for some α, and $C_1\mu\alpha$ is Σ-solvable, since $C_1\mu \wedge C_2\mu$ is Σ-solvable by definition 6.7. Therefore, $w \in GrLW(t_1,C_1)$. A symmetric argument holds for $w \in GrLW(t_2,C_2)$.

($\supseteq$) Now, take any w such that $w \in GrLW(t_1,C_1)$ and $w \in GrLW(t_2,C_2)$. Then $w = t_1\alpha$ for some α, and $w = t_2\beta$ for some β. Assume WLOG[18] that C_1 and C_2 are variable disjoint. Thus, $w = t_1(\alpha\beta) = t_2(\alpha\beta)$. Hence, t_1 and t_2 unify with an MGU, say μ. Thus, $(\alpha\beta) = \mu\sigma$ for some possibly empty σ. Since $w = t_1\mu\sigma \in GrLW(t_1,C_1)$, $C_1\mu\sigma$ must be Σ-solvable.

---

18) WLOG = without loss of generality

The same holds for $C_2\mu\sigma$. Thus, the conjunct $C_1\mu\sigma \wedge C_2\mu\sigma$ is Σ-solvable. Thus, $w \in GrLW(t_1\mu, C_1\mu \wedge C_2\mu)$. ■■

Lemma 6.3 (Σ-consistency of Ground Paths) *Given a constraint theory Σ, a ground world path P is Σ-consistent if and only if*

$$\Sigma \models constraint(P)$$

that is:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{n-1},P_n)$$

Proof: Since all P_i are ground, the definition of Σ-consistency requires

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{i-2},P_{i-1}) \rightarrow K(P_{i-1},P_i)$$

for all i. This is:

$$\Sigma \models K(P_0,P_1)$$

$$\Sigma \models K(P_0,P_1) \rightarrow K(P_1,P_2)$$

$$\Sigma \models K(P_0,P_1) \wedge K(P_1,P_2) \rightarrow K(P_2,P_3)$$

...

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{n-2},P_{n-1}) \rightarrow K(P_{n-1},P_n)$$

This is equivalent to:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{n-1},P_n)$$ ■■

The next lemma states that there can only be one access path to each world. For instance, it is not possible to have the two paths $0 \rightarrow sk_1 \rightarrow sk_2$ and $0 \rightarrow sk_3 \rightarrow sk_2$ in the course of a deduction. In other words, once the final world of a path is known, the whole path is determined.

Scherl proved a similar property of the *Least Herbrand Model* of Σ. Freely phrased, a ground literal is true in the Least Herbrand Model if and only if it is true in all models. Thus, saying $K(t_1,t_2)$ is true in the Least Herbrand Model, is saying $\Sigma \models K(t_1,t_2)$. If the accessibility of t_2 from t_1 in the Least Herbrand Model is represented as an edge in a graph such that the vertexes are ground worlds, then the graph forms a tree with 0 as the root [Scherl 92].

Lemma 6.4 (Ground Path Identity) *Given two Σ-consistent paths P and Q such that both are ground, and last(P) = last(Q), then P = Q.*

Proof: Suppose $P \neq Q$. Since both paths are ground and Σ-consistent, lemma 6.3 applies. Thus:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{n-1},P_n)$$

and

$$\Sigma \models K(Q_0,Q_1) \wedge \ldots \wedge K(Q_{m-1},Q_m).$$

By the hypothesis, $P_n = Q_m$. Going from right to left, there must be a first term in P that differs from its counterpart in Q. Formally, there must be an i such that $P_i \neq Q_j$, where $j=i+m-n$ and $P_{i+k} = Q_{j+k}$ for all $k \in \{1,\ldots,n-i\}$. Thus, $\Sigma \models K(P_i,t) \wedge K(Q_j,t)$, where $t = P_{i+1} = Q_{j+1}$.

Now, the clause that makes $K(P_i,t)$ true, cannot be the reflexivity axiom. As covered in Section IV.C, whenever unification substitutes the neutral element '1' into the path, the '1' will be removed right away, because the equality theory states $\forall w \;\; w \rightarrow 1 \equiv w$. Thus, there is no pair $P_x \rightarrow P_{x+1}$ in the path such that $P_x = P_{x+1}$.

Similar reasoning holds for symmetry and the inverse element $(\,)^{-1}$. Now consider an application of transitivity, where unification substitutes the substring of another path into the first path, as in $P_0 \rightarrow P_1 \rightarrow (Q_k \rightarrow \ldots \rightarrow Q_{k+i}) \rightarrow P_3 \rightarrow \ldots$ This substitution is possible, *because* $K(P_1,Q_{k+i})$ follows from transitivity. But P_1 and Q_{k+i} are *not* immediate neighbors in the path. Thus, there is no pair $P_x \rightarrow P_{x+1}$ in the path such that $K(P_x,P_{x+1})$ relies on the transitivity axiom in Σ.

Therefore, the only remaining clauses in Σ, that can possibly make $K(P_i,t)$ and $K(Q_j,t)$ true, are the seriality clause and the skolem clauses. In order for a K literal to be entailed by Σ, it must match a right-hand side of one of the clauses in Σ. Recall that all clauses in Σ are definite, so the have exactly one positive literal, which makes up the right-hand side of a clause. The positive K literals in the seriality clause and in the skolem clauses are of the form K(w,f(w)), where f is some function name. However, the translation procedure into WPL ensures no two clauses in Σ use the same function f in their positive K literal. The translation introduces a new function, different from all other functions, at every skolemization step.

Thus, since t is ground, both $K(P_i,t)$ and $K(Q_j,t)$ match the positive literal of the same clause, which is of the form K(w,f(w)). But then P_i must be equal to Q_j, thus contradicting our initial assumption that $P_i \neq Q_j$. ■■

Now we have all the machinery available to prove that unification of paths is equivalent to the conjunction of constraints, as far as possible final worlds are concerned. This proof will complete the second section of this chapter.

Theorem 6.3 *Given two Σ-consistent paths, P1 and P2, and two constrained world terms, t_1 and t_2, along with their constraints, C_1 and C_2, such that:*

- *GrLW(P1) = GrLW(t_1,C_1)*
- *GrLW(P2) = GrLW(t_2,C_2)*

then GrLW(S) = GrLW($t_1\mu$, $C_1\mu \wedge C_2\mu$), where S = {P1σ | σ is an MGU of P1 and P2} and μ is the MGU of t_1 and t_2.

Proof: By lemma 6.2, GrLW($t_1\mu$, $C_1\mu \wedge C_2\mu$) = GrLW(t_1,C_1) ∩ GrLW(t_2,C_2). Thus, it suffices to show that:

- GrLW(S) = GrLW(t_1,C_1) ∩ GrLW(t_2,C_2)

(⊆) Take any member w of any GrLW(P1σ) such that σ is an MGU of P1 and P2. Then w is *last*(P1σμ) for some μ such that P1σμ is Σ-consistent and ground (definition 6.6). Thus, w is also a member of GrLW(P1), with substitution $\alpha=\sigma\mu$. But then, by the theorem's hypothesis, w is also a member of GrLW(t_1,C_1).

Since P1σ = P2σ, a similar argument shows that w is also in GrLW(t_2,C_2).

(⊇) Suppose w ∈ GrLW(t_1,C_1) ∩ GrLW(t_2,C_2), but w ∉ GrLW(P1σ) for any σ such that σ is an MGU of P1 and P2. Then w ∈ GrLW(P1) and w ∈ GrLW(P2). Thus, there exist an α and a β such that w = *last*(P1α) = *last*(P2β). Therefore, by lemma 6.4, P1α=P2β. WLOG assume, P1 and P2 are variable disjoint. Then there must exist an MGU σ of P1 and P2 such that (αβ) = σμ for some possibly empty μ. Thus, P1α = P1σμ and w = *last*(P1σμ) and P1σμ=P1α is Σ-consistent. But then w ∈ GrLW(P1σ), which contradicts our assumption. ■■

C. Unification as a Test For Σ-Solvability

Whenever two constraints are combined during the course of deduction in RML Constraint Logic, the Σ-solvability of the joint constraint has to be tested. A constraint is

Σ−solvable, if there is a ground instance of that constraint such that it is entailed by Σ. Unfortunately, the results of a test for Σ-solvability are not reused in subsequent tests of new constraints that are derived from existing ones. Moreover, the constraints get longer and longer as the deduction progresses. Thus, the method is quite inefficient.

All methods that work with path unification of some sort [Jackson, Reichgelt 87; Auffray, Enjalbert 89; Ohlbach 88; Frisch, Scherl 91] have an important edge over the constraint logic method. They do not require an explicit test for Σ-solvability. Unification takes care of it. World paths do not unify, unless their combination is Σ-solvable. What is more, by instantiating world variables in the path, the implicit Σ-solvability test works incrementally. The role of unification can be interpreted as to instantiate variables just enough to ensure this Σ-solvability, but to delay the actual choice of ground instances for as long as possible.

In this section we will demonstrate that these properties apply to our World Path Logic. In particular, we will prove that all paths which can possibly occur during the course of a deduction, are Σ-solvable. Our approach works along the notion of Σ-consistency, as defined in definition 6.1. In order to prove that all paths are Σ-solvable, it suffices to prove that:

(a) the initial paths resulting directly from the translation into WPL are Σ-consistent
(b) instantiating a path with a most general unifier of two paths preserves Σ-consistency
(c) every Σ-consistent path corresponds to a Σ-solvable constraint

Property (a) has already been proven as lemma 6.1. The proof of property (b) is quite long and tedious; therefore, we will do (c) first.

Theorem 6.4 (Σ-consistency ⇒ Σ-solvability) *Given a world path P such that P is Σ-consistent, then constraint(P) is Σ-solvable.*

Proof: *constraint*(P) is equal to $K(P_0,P_1) \wedge \ldots \wedge K(P_{n-1},P_n)$. By definition, this conjunction is Σ-solvable, iff there exists a substitution μ such that $(K(P_0,P_1) \wedge \ldots \wedge K(P_{n-1},P_n))\mu$ is ground and

$$\Sigma \models (K(P_0,P_1) \wedge \ldots \wedge K(P_{n-1},P_n))\mu \tag{6.13}$$

Construct μ recursively as follows, where f( ) is the function used in the seriality axiom K(w,f(w)) in Σ:

$\mu_0 = \{\}$

$\mu_i = \mu_{i-1} \circ \{f(P_{i-1})/P_i\}$, if P_i is a variable

$\mu_i = \mu_{i-1}$, otherwise

$\mu = \mu_n$

Thus, μ substitutes every variable term P_i with $f(P_{i-1})$ ensuring that P_i is accessible from P_{i-1} by seriality. The point of having a recursive definition of μ is to cover the case, where the path has multiple variables in a row, thus making sure only ground terms are substituted for variables. Thus, $(K(P_0,P_1) \wedge \ldots \wedge K(P_{n-1},P_n))\mu$ is ground as far as world variables are concerned.

The conjunction may still contain some non-world variables, but then we can expand μ to substitute them with any domain element, say 'a', without affecting the accessibility at all.

The proof of (6.13) is by contradiction. (6.13) is equivalent to

$$\Sigma \models K(P_0,P_1)\mu \wedge \ldots \wedge K(P_{n-1},P_n)\mu \tag{6.14}$$

Suppose (6.14) does not hold. Then there must be leftmost K literal $K(P_{i-1},P_i)\mu$ which is not entailed by Σ. Thus:

$$\Sigma \not\models K(P_{i-1},P_i)\mu \tag{6.15}$$

[19]

but

$$\Sigma \models K(P_0,P_1)\mu \wedge \ldots \wedge K(P_{i-2},P_{i-1})\mu \tag{6.16}$$

P_i can either be a variable or not. Suppose it is, then $K(P_{i-1},P_i)\mu = K(P_{i-1},f(P_{i-1}))$. But Σ entails this literal by seriality, thus contradicting (6.15).

Now suppose P_i is not a variable. Then, by Σ-consistency of P,

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{i-2},P_{i-1}) \rightarrow K(P_{i-1},P_i) \tag{6.17}$$

The conjunction in (6.16) is just an instance of the antecedent in (6.17). Thus $\Sigma \models K(P_{i-1},P_i)\mu$ which is a contradiction to (6.15). ■■

It remains to be shown that instantiating a path with a most general unifier preserves Σ-consistency. First, we will prove this for a substitution σ on P such that σ is the MGU of this path P and some other path. Then this result will be extended to show that σ can be an MGU of any two paths, not necessarily including P. The following lemma is needed for the proof:

---

19) $\not\models$ is meant to denote the negative $\models$ operator. $\alpha \not\models \beta$ reads: α does not entail β

Lemma 6.5 (No New World Variables in Skolem Arguments) *Let P be a world path and P_i a function term in P. Then all world variables occurring in P_i also occur in the P_i's prefix $P_0 \rightarrow \ldots \rightarrow P_{i-1}$.*

Proof: It follows from the translation function that this property holds for all initial paths resulting from translation. Moreover, if it holds for P, then it must also hold for Pσ. This follows from the properties of substitution. Thus, the property is preserved over the course of a deduction. ∎∎

Theorem 6.5 (Σ-consistency Preservation, Part 1) *Given a constraint theory Σ, two Σ-consistent paths P and Q, and an MGU σ of P and Q, then Pσ will also satisfy the property of Σ-consistency.*

Proof: WLOG assume P and Q are variable disjoint. Let us also assume WLOG that σ = {t1/x1,...} such that no variable x1,...,xn occurs in any of the terms t1...tn. We will now redo the substitution in P and Q step by step from left to right.

Let prefix(n,P) denote the first n+1 terms of P, i.e. prefix(n,P) = $P_0 \rightarrow \ldots \rightarrow P_n$. Then the following property holds for P, Q, k, and σ (<u>induction hypothesis</u>):

- prefix(k−1,P) = prefix(k−1,Q)
- prefix(k−1,P) does not contain any variable that σ substitutes
- P and Q are Σ-consistent

This property certainly holds for the <u>base case</u> k = 1, the MGU σ, and the initial paths P and Q, which are Σ-consistent by lemma 6.1. As the <u>inductive step</u> will now show, we can always pick a non-empty μ, $\mu \subseteq \sigma$, and apply it to P and Q such that the induction hypothesis holds for P', Q', k', and σ', where

- P' = P μ
- Q' = Q μ
- $\sigma' = \sigma - \mu$

The proof is basically an induction on the cardinality of σ. Since σ is reduced in size at every induction step, it will eventually be empty. This means that the complete substitution will have been performed. Thus, the resulting P' is Pσ (P and σ from the base case), and it is still consistent with Σ which was to be proven.

We will now complete the proof with the details of the inductive step. Consider the world terms P_k and Q_k and distinguish six cases:

(i) Both terms are variables and σ contains a substitution μ such that $\mu = \{P_k/Q_k\}$ or $\mu = \{Q_k/P_k\}$. Perform this substitution, i.e. let P' = $P\mu$, and let Q' = $Q\mu$. The induction hypothesis then holds for P', Q', k'=k+1, and $\sigma' = \sigma - \mu$. ∎

(ii) Both terms are non-variables, i.e. a function (we consider constants zero-ary functions). Then the function must be the same in P_k as in Q_k, otherwise P and Q would not unify. Let us look at possible variables in the argument terms to that function. Concerning the world variables, it follows from lemma 6.5 that they also occur in prefix(k,P). And by the induction hypothesis, σ does not substitute them. Thus, all world variables in P_k, if any, equal their respective counterpart in Q_k.

As to the non-world variables, do all substitutions μ in σ pertaining to them and remove those μ from σ, yielding P',Q',σ' respectively.

In order to show that P' is Σ-consistent, we need to have $\Sigma \models K_1 \wedge \ldots \wedge K_{i-1} \rightarrow K_i$ for every non-variable world term P'_i in P'. Since we did not substitute world-terms, P'_i is a non-variable term if and only if P_i is a non-variable. Thus, all the clauses that need to be entailed by Σ for P', have corresponding clauses in P, of which they are instances. So, if Σ entails

$$K(P_0,P_1) \wedge \ldots \wedge K(P_{i-2},P_{i-1}) \rightarrow K(P_{i-1},P_i)$$

then Σ also entails the instance

$$(\, K(P_0,P_1) \wedge \ldots \wedge K(P_{i-2},P_{i-1}) \rightarrow K(P_{i-1},P_i)\,)\,\mu$$

which is equal to

$$K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{i-2},P'_{i-1}) \rightarrow K(P'_{i-1},P'_i)$$

A similar argument holds for Q'. Thus, the hypothesis holds for P', Q', $\sigma'=\sigma-\mu$, and k'=k+1. ∎

(iii) One term is a variable, the other term is not. WLOG assume, P_k is the variable. Then there must be a substitution $\mu = \{ t/P_k \}$ in σ such that t is an instance of Q_k. Let P' = $P\mu$ and observe that

$$K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{k-2},P'_{k-1}) \rightarrow K(P'_{k-1},P'_k) \qquad (6.18)$$

is just an instance of

$$K(Q_0,Q_1) \wedge \ldots \wedge K(Q_{k-2},Q_{k-1}) \rightarrow K(Q_{k-1},Q_k) \qquad (6.19)$$

because prefix(k−1,P) = prefix(k−1,Q) and σ does not effect prefix(k−1,P). Since (6.19) is entailed by Σ, so is (6.18).

Next consider the terms P'_i in P', such that $i > k$. We need to show that $\Sigma \models K_1 \wedge \ldots \wedge K_{i-1} \rightarrow K_i$ for every such non-variable world term P'_i. Note that P'_i is a non-variable if and only if P_i is. $\mu = \{ t/P_k \}$ does not substitute any P_i, because every P_i is different from P_k. This follows from the *unique prefix property* (theorem 4.1). As in case 2, if Σ entails

$$K(P_0,P_1) \wedge \ldots \wedge K(P_{i-2},P_{i-1}) \rightarrow K(P_{i-1},P_i)$$

then Σ also entails the instance

$$(K(P_0,P_1) \wedge \ldots \wedge K(P_{i-2},P_{i-1}) \rightarrow K(P_{i-1},P_i))\ \mu$$

which is equal to

$$K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{i-2},P'_{i-1}) \rightarrow K(P'_{i-1},P'_i)$$

Thus, P' has the property of Σ-consistency. The induction hypothesis holds for P', $Q'=Q\mu=Q$, $k'=k$, and $\sigma'=\sigma-\mu$.

Recall that P'_k and Q_k are not necessary equal. Both are function terms, however. And if they are actually not identical, then case (ii) will apply at the next round. ∎

(iv) <u>Reflexivity</u> holds, and the neutral element '1' is substituted for the variable P_k or Q_k. WLOG assume it is P_k, which is then deleted from the world path under application of the equality theory for reflexivity, $\forall w\ \ w \rightarrow 1 \equiv w$. Thus, $P'_j = P_j$ for all $j < k$, and $P'_j = P_{j+1}$ for all $j \geq k$. Consider a skolem function term P'_j, such that $j \geq k$. We need to show that

$$\Sigma \models K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j) \qquad (6.20)$$

Σ-consistency of P tells us that

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{j-1},P_j) \rightarrow K(P_j,P_{j+1}) \qquad (6.21)$$

Since P_k is a variable, we can instantiate it to P_{k-1}:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{k-1},P_{k-1}) \wedge K(P_{k-1},P_{k+1}) \wedge \ldots \wedge K(P_{j-1},P_j) \rightarrow K(P_j,P_{j+1}) \qquad (6.22)$$

Note that the reflexivity axiom $\forall w\ K(w,w)$ is part of Σ. Thus, we can resolve the literal $K(P_{k-1},P_{k-1})$ in (6.22) away, yielding:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{k-1},P_{k+1}) \wedge \ldots \wedge K(P_{j-1},P_j) \rightarrow K(P_j,P_{j+1}) \qquad (6.23)$$

This is actually identical to:

$$\Sigma \models K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{k-1},P'_k) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j) \qquad (6.24)$$

And (6.24) again is equal to (6.20). Thus, P' preserves Σ-consistency. The induction hypothesis holds for P', $Q'=Q$, $k'=k$, $\sigma'=\sigma-\mu$. ∎

(v) <u>Symmetry</u> holds, and the inverse element P_k^{-1} is substituted for the variable P_{k+1} (or Q_{k+1}, WLOG assume it is P_{k+1}). Then P_k and P_{k+1} can both be deleted from the world path under application of the equality theory for symmetry,

$$\forall w,w' \quad w \rightarrow w' \rightarrow w'^{-1} \equiv w$$

Thus, $P'_j = P_j$ for all $j < k$, and $P'_j = P_{j+2}$ for all $j >= k$. Consider a skolem function term P'_j, such that $j >= k$. We need to show that

$$\Sigma \models K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j) \qquad (6.25)$$

Recall that P is Σ-consistent. Thus:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_j,P_{j+1}) \rightarrow K(P_{j+1},P_{j+2}) \qquad (6.26)$$

Since P_{k+1} is a variable, we can instantiate it to P_{k-1}:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{k-1},P_k) \wedge K(P_k,P_{k-1}) \wedge K(P_{k-1},P_{k+2}) \wedge \ldots \wedge K(P_j,P_{j+1}) \rightarrow K(P_{j+1},P_{j+2}) \qquad (6.27)$$

Note that the symmetry axiom $\forall w,w' \; K(w,w') \rightarrow K(w',w)$ is part of Σ. Thus, we can resolve the literal $K(P_k,P_{k-1})$ in (6.27) away, yielding:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{k-1},P_k) \wedge K(P_{k-1},P_{k+2}) \wedge \ldots \wedge K(P_j,P_{j+1}) \rightarrow K(P_{j+1},P_{j+2}) \qquad (6.28)$$

Next, we want to get rid of $K(P_{k-1},P_k)$. Suppose P_k is a *non-variable*. Then, by Σ-consistency of P,

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{k-2},P_{k-1}) \rightarrow K(P_{k-1},P_k) \qquad (6.29)$$

Now suppose P_k is a *variable*. In this case we can just instantiate P_k to $f(P_{k-1})$ and use the seriality axiom, which is part of Σ:

$$\forall w \quad K(w,f(w)) \qquad (6.30)$$

Either way, (6.29) or (6.30), we can resolve $K(P_{k-1},P_k)$ out of (6.28), yielding:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{k-1},P_{k+2}) \wedge \ldots \wedge K(P_j,P_{j+1}) \rightarrow K(P_{j+1},P_{j+2}) \qquad (6.31)$$

This is actually identical to:

$$\Sigma \models K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{k-1},P'_k) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j) \qquad (6.32)$$

And (6.32) again is equal to (6.25). Thus, P' (and Q') preserve Σ-consistency. The induction hypothesis holds for P', Q', $\sigma'=\sigma-\mu$, and $k'=k$. ■

(vi) <u>Transitivity</u> holds, and the variable P_k is being substituted by a subsequence of Q. $\mu = \{ (Q_k \rightarrow \ldots \rightarrow Q_{k+i})/P_k \}$ is in σ. Using the equality theory axiom for transitivity,

$$\forall w,w',w'' \quad w \rightarrow w' \rightarrow w'' \equiv w \rightarrow (w' \rightarrow w'')$$

the path $\qquad P_0 \rightarrow \ldots \rightarrow P_{k-1} \rightarrow (Q_k \rightarrow \ldots \rightarrow Q_{k+i}) \rightarrow P_{k+1} \rightarrow \ldots \rightarrow P_n$

can be rewritten as

$$P_0 \rightarrow \ldots \rightarrow P_{k-1} \rightarrow Q_k \rightarrow \ldots \rightarrow Q_{k+i} \rightarrow P_{k+1} \rightarrow \ldots \rightarrow P_n$$

Thus, $P'_j = P_j = Q_j$ for all $j<k$, $P'_j = Q_j$ for all j, such that $k \leq j \leq k+i$, and $P'_j = P_{j-i}$ for all $j > k+i$. Consider a skolem function term P'_j, such that $j \leq k+i$. Since prefix(k+i,P') = prefix(k+i,Q) and Q is consistent with Σ, it follows that

$$\Sigma \models K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j) \qquad (6.33)$$

We need to show, that (6.33) also holds for skolem function terms P'_j, such that $j > k+i$. As previously stated, $P'_j = P_{j-i}$, and, since P is Σ-consistent, we know:

$$\Sigma \models K(P_0,P_1) \wedge \ldots \wedge K(P_{j-i-2},P_{j-i-1}) \rightarrow K(P_{j-i-1},P_{j-i}) \qquad (6.34)$$

This can be rewritten as:

$$\Sigma \models K(Q_0,Q_1) \wedge \ldots \wedge K(Q_{k-1},P_k) \wedge K(P_k,P'_{k+i+1}) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j) \quad (6.35)$$

Note that the transitivity axiom

$$\forall w,w',w'' \quad K(w,w') \wedge K(w',w'') \rightarrow K(w,w'')$$

is part of Σ. Thus, the following is also deducible:

$$\Sigma \models K(Q_{k-1},Q_k) \wedge \ldots \wedge K(Q_{k+i-1},Q_{k+i}) \rightarrow K(Q_{k-1},Q_{k+i}) \qquad (6.36)$$

Since P_k in (6.35) is a variable, the literals $K(Q_{k-1},P_k)$ in (6.35) and $K(Q_{k-1},Q_{k+i})$ in (6.36) are unifiable with $\{ Q_{k+1}/P_k \}$, and therefore, (6.35) and (6.36) resolve to:

$$\Sigma \models K(Q_0,Q_1) \wedge \ldots \wedge K(Q_{k+i},P'_{k+i+1}) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j) \quad (6.37)$$

This, however, can simply be rewritten to:

$$\Sigma \models K(P'_0,P'_1) \wedge \ldots \wedge K(P'_{j-2},P'_{j-1}) \rightarrow K(P'_{j-1},P'_j)$$

which means that P' has the property of Σ-consistency. The induction hypothesis holds for P', $Q' = Q\mu = Q$, $\sigma' = \sigma - \mu$, and $k' = k+i+1$.

This concludes the proof of the Σ-consistency preservation (part 1) property. ∎∎

Theorem 6.6 (Σ-consistency Preservation, Part 2) *Given a constraint theory Σ, three Σ-consistent paths P, Q, R, and an MGU σ of P and Q, then $R\sigma$ will also satisfy the property of Σ-consistency.*

Proof: The key to this proof is the observation that for every two paths P, R, resulting from the translation into World Path Logic, there is an integer $k \geq 0$, such that

- prefix(k,P) = prefix(k,R)

and

- $\forall i,j_{:\, i>k\, ,\, j>k} \quad P_i \neq R_j$

In other words, two paths are identical up to a certain point in the path, and then they are completely different. This follows from the structure of modal logic and the translation function. Consider any two literals and the modal operators, that they are in the scope of. Since scopes are nested structures, the operators that have both literals in their scope, must all be outside of the operators which have only one of the literals in their scope. The common outside operators account for the common prefix of the two world paths.

Now consider the paths R, which is given to be Σ-consistent, and $P\sigma$, which is Σ-consistent by theorem 6.5. Recall the method of the proof of theorem 6.5. Two Σ-consistent paths were unified step by step from left to right, preserving Σ-consistency of the full paths at every single step. Since P and R are identical on the first k elements, a substitution of σ into R is in effect the same as a partial unification of R and $P\sigma$ from left to right for the first k elements. In the previous proof of theorem 6.5 we also unified two paths stepwise from left to right, and each partial unification step preserved Σ-consistency of the whole path. Thus, following the approach of the previous proof, the resulting path of the partial unification, which is $R\sigma$, must still be Σ-consistent. ■■

D. Summary

When the same modal logic formula is translated into both World Path Logic and RML/CL, then there is a close similarity between the WPL terms and the RML/CL terms. In particular, for every WPL term there is corresponding RML/CL term such that (a) the WPL term's path matches the constraint of the RML/CL term and (b) the WPL term equals the RML/CL term, except that the WPL term uses paths where the RML/CL term has simple world terms. Furthermore, Section A proved that WPL terms have the same ground instances as their corresponding RML/CL terms.

Section B related the world paths to RML constraints. In particular, we showed that a world path resulting directly from the translation represents the same set of worlds as its corresponding constraint in RML/CL. Applying the results from Section A, this means that they are also equal with respect to their ground instances. Moreover, we proved that this identity is preserved over a deduction step. In other words, unification of paths is equivalent to the conjunction of constraints as far as possible final worlds and ground instances are concerned.

Since the ground instances are equal to begin with and throughout the deduction, we have a ground refutation in RML/CL if and only we have a ground refutation in WPL. In addition, Section C confirmed that the test for Σ-solvability of the RML constraints is obsolete in WPL, because two paths unify if and only if the conjunct of the corresponding RML constraints is Σ-solvable. Thus, the soundness and completeness results from RML/CL deduction carry over to WPL deduction.

Deduction methods are usually expected to be sound and refutation-complete[20]. This means, all derivable formulas follow from the set of premises (soundness), and if the set of premises is unsatisfiable, then there exists a deduction ending in 'false' or, in a clausal resolution system, the empty clause (completeness). RML/CL deduction was shown to be sound and complete in [Scherl 92]. As we have just showed, there is a direct correspondence between deduction in WPL and in RML Constraint Logic, to the effect that every deduction step in WPL can be simulated in RML/CL and vice versa. Therefore, deduction in World Path Logic is sound and complete, too.

Now that we have obtained a theoretical understanding of deduction in World Path Logic, we are ready to approach subsumption in the next chapter.

20) as opposed to 'deduction complete'. Most methods cannot deduce a tautological clause like $b \vee \neg b$, where b is a new literal not occurring in the premises, although $b \vee \neg b$ is entailed by the premises [Wos 93].

VII. SUBSUMPTION

Chapter VI concluded with the statement that deduction in WPL is sound and complete. While soundness and completeness are essential issues, practical implementations of automated theorem provers have to face a broader variety of problems. For instance, it is nice to know that a refutation will eventually be found, if the set of premises is unsatisfiable, but how long is eventually'? If the set of premises is in fact satisfiable, then the system might search forever for a refutation proof. This undecidability of FOPL and extended first order logics like modal logic is a fundamental problem of computer science which cannot be overcome even by developing more sophisticated algorithms. Nevertheless, designing the proof search more efficient helps to ease the practical consequences of this theoretical problem.

There are several ways to improve the efficiency of this search. A clausal resolution based theorem prover usually generates a large amount of clauses, most of which later turn out not to be needed for the proof. Also, at any deduction step, the theorem prover has to decide which clauses out of the large search space of given and derived clauses to resolve next. Resolution strategies are concerned with a good choice of clauses to be resolved next, in order to obtain a proof fast. Other methods are aimed to keep the set of clauses to choose from small. This is what subsumption is designed to do. The basic idea of subsumption is to remove those clauses that can be derived from a single other clause. If clauses B and C resolve to clause D, where B follows directly from clause A, then D can also be derived from A and C. Thus, B is unnecessary and redundant. We say A subsumes B. One can distinguish three different types of subsumption [Wos 93]:

(i) Forward Subsumption. Once a new clause B has been derived, it is compared against the set of existing clauses to see, if there is a clause A among them such that B is subsumed by A. Then B is dropped.

(ii) Backward Subsumption. Once a new clause B has been derived, it is compared against the set of existing clauses to see, if there is a clause A among them such that B subsumes A. Then the old clause A is replaced by B.

(iii) Ancestor Subsumption. Once a new clause B has been derived, it is compared against the set of existing clauses in order to see, if there is a clause A among them such that A $\equiv$ B. In terms of subsumption, A $\equiv$ B means: A subsumes B, and B subsumes A. Then the clause with the shorter derivation path is kept, and the other one is removed. The motivation behind this is to obtain short proofs.

Ancestor subsumption is somewhat different from the other two kinds, in that it involves checking the length of derivation paths. This, however, is not within the scope of our treatment. We are only concerned about 'simple' subsumption detection as in (i) and (ii). Here is a more formal definition of subsumption:

Definition 7.1 (Subsumption) *A formula α subsumes a formula β, iff $\forall^* \alpha \models \forall^* \beta$, where $\forall^*$ denotes the universal closure of a formula such that all free variables are universally quantified.*

The statement 'α subsumes β' can be read as: α implies β, α entails β, or α is more general than β. The universal closure reflects the understanding that all free variables are meant to be universally quantified. Suppose α and β are part of a given and/or derived set of formulas such that all formulas hold jointly. Then taking the universal closure of the conjunction of all formulas is equivalent to universally closing every single formula, because $\forall x\ (\alpha(x) \land \varphi(x)) \equiv (\forall x\ \alpha(x)) \land (\forall x\ \varphi(x))$. Therefore, the universal quantifier can be applied to every single formula like α and β.

Note that definition 7.1 is quite general. It does not restrict α and β to be clauses. Also, the logic language under consideration is not specified. Subsumption is of practical relevance only in deduction systems. Thus, when we speak of subsumption in modal logic, we mean subsumption in the language that we are doing modal logic proofs in. As pointed out in Chapter II, some deduction methods construct their proofs directly in modal logic, while other techniques prefer an indirect approach. They translate modal logic formulas into another language and then try to do the proof in that target language. With the World Path Language presented in Chapter IV, we follow this direction. Thus, the problem of subsumption in modal logic, i.e. in modal logic proof systems, reduces to the problem of subsumption in the target language.

There is also another reason for dealing with subsumption at the target language level. Using subsumption checks makes sense only if the possible benefits outweigh the costs for the subsumption tests. While subsumption is relatively easily determined among clauses, it can be quite expensive in more complex structured formulas. Unfortunately, modal logic formulas cannot always be converted into an equivalent clausal form such that clauses do not contain any conjunctions. Just consider the example $\varphi = p \vee \Diamond(q \wedge r)$. Conversion to clausal form would mean breaking up the inner conjunction such that q and r end up in different clauses. But then there is no way to represent the fact that q and r pertain to the *same* world.

Therefore, the rest of this chapter will be devoted to examining subsumption among World Path Logic clauses. Nevertheless, it is still possible to check if one modal logic formula, say α, subsumes another one, say β. Just translate $\alpha \wedge \neg\beta$ into WPL and search for a refutation proof.

The rest of this chapter is organized as follows: Section A covers subsumption for unit clauses, i.e. clauses that consist of just one literal. Section B extends those results to non-unit clauses. Then we will present an algorithm for subsumption detection and prove its correctness in Section C. Finally, this chapter closes with a summary of the results in Section D.

A. Unit Clauses

This section treats subsumption for WPL unit clauses. Unit clauses consist of exactly one literal. This makes subsumption relatively easy to determine, because one literal, say L1, obviously entails another one, say L2, only if the predicate is the same in both L1 and L2, and if either both are negative or both are positive.

By definition 7.1, L1 subsumes L2 if and only if $\forall^* L1 \models \forall^* L2$. From the semantics of universal quantification, $\forall^* L$ is true, if all of L's ground instances are true. In this respect, we can treat a universally closed literal with variables as a scheme standing for all of its ground instances. So, if G(L) denotes the set of all ground instances of L, then L1 subsumes L2 iff $G(L1) \supseteq G(L2)$.

For WPL, however, this is not entirely true. A literal like $p(0 \rightarrow w)$ does *not* entail all ground instances such that we can instantiate w with any world. The variable w stands only for those worlds that are accessible from world 0. The quantification is in fact constrained. Remember that WPL expressions are a representation of modal logic. As for our example,

$p(0 \rightarrow w)$ represents the modal logic term $\Box p$. And $\Box p$ does not require p to hold in *all* worlds, but only in those that are accessible from the current world, which is 0.

Let us extend the previous example, and find out which worlds w may be instantiated to. Suppose the set of modal logic sentences under consideration is $\{ \Box p, \Box \Diamond \neg p, \Diamond q \}$, and the modal logic is serial only (system KD). The translation into WPL yields: $\{p(0 \rightarrow w), \neg p(0 \rightarrow w_2 \rightarrow sk_1), q(0 \rightarrow sk_2)\}$. Then the only possible instantiations for w are sk_2 and f(0), where f( ) is the function used in the seriality axiom in the constraint theory Σ. The literal $p(0 \rightarrow sk_1)$ is not a ground instance that $\forall^* p(0 \rightarrow w)$ stands for, because sk_1 is not necessarily accessible from 0.

Definition 6.5 defines the Σ-ground instances of a WPL literal, and $\Sigma gr(\,)$, the set of Σ-ground instances, accordingly. Given a literal L, $\Sigma gr(L)$ is the set of all *LAST*($L\sigma$), such that $L\sigma$ is ground and Σ-consistent. Σ-consistency makes sure that only accessible worlds are instantiated, and *LAST*( ) (see definition 6.2) replaces all world paths by their last element. Therefore, we can state the following lemma:

Lemma 7.1 *Given two WPL literals L1 and L2, L1 subsumes L2 iff* $\Sigma gr(L1) \supseteq \Sigma gr(L2)$.

Our goal is to show that a subsumption test in World Path Logic works just like in regular first order predicate logic. That is: L1 subsumes L2, if there exists a substitution σ such that $L2 = L1\sigma$. In other words, L1 subsumes L2, if L2 is an instance of L1. However, substitution in WPL is more complex than in FOPL. WPL substitution effects not merely regular variables, but also world paths, and can thus contain special elements as described in the section on world path unification. These are, for instance, the neutral element in the case of a reflexive logic, inverse elements, if the logic is symmetric, and nested subpaths in the case of transitivity. Regardless of these differences, the subsumption test method is basically the same: it means finding a substitution σ such that $L1\sigma = L2$.

Theorem 7.1 (WPL Literal Subsumption) *Given two WPL literals L1 and L2 such that L1 and L2 result from modal logic translation or are derived in the course of a deduction, then L1 subsumes L2 if and only if* $\exists\sigma\ L1\sigma = L2$.

The restriction on L1 and L2 has no practical relevance, because for all practical purposes of subsumption, there is no other source where WPL literals can stem from. As for the proof however, the restriction ensures the Σ-consistency of L1 and L2.

Proof: By lemma 7.1, L1 subsumes L2 iff $\Sigma gr(L1) \supseteq \Sigma gr(L2)$. Thus, it suffices to show:

$$\Sigma gr(L1) \supseteq \Sigma gr(L2) \text{ iff } \exists\sigma\ L1\sigma = L2$$

"if": Pick any Σ-ground instance of L2, say L2'. Then L2' = $L2\mu$ for some μ. Thus, L2' = $L2\mu$ = $L1\sigma\mu$ is also a Σ-ground instance of L1. ■

"only if": First, we will prove that $\Sigma gr(L2)$ is not empty. For if $\Sigma gr(L2)$ were empty, then $\Sigma gr(L1) \supseteq \Sigma gr(L2)$ would hold trivially, regardless of the existence of a substitution σ.

L2 is Σ-consistent, because it either results directly from modal logic translation, then it is Σ-consistent by lemma 6.1 (initial Σ-consistency). Or it was created in the course of a deduction, then it is Σ-consistent by theorem 6.6 (Σ-consistency preservation). Lemma 7.2, which follows right after this proof, states that all Σ-consistent literals have at least one Σ-ground instance. Thus, $\Sigma gr(L2) \neq \emptyset$. Now suppose $\Sigma gr(L1) \supseteq \Sigma gr(L2)$, but there exists no σ such that $L1\sigma = L2$. We will show that this assumption leads to a contradiction.

Case 1: L1 and L2 do not unify. Then L1 and L2 have no Σ-ground instance in common, because a common ground instance of L1 and L2 would mean the existence of a unifier. But since $\Sigma gr(L2) \neq \emptyset$, L2 has at least one Σ-ground instance that is not a Σ-ground instance of L1. Thus, $\Sigma gr(L1) \supseteq \Sigma gr(L2)$ cannot hold. ■

Case 2: L1 and L2 do unify, but the most general unifier substitutes a non-variable term, say t, for a variable in L2, say x (otherwise $L1\sigma = L2$ would hold for some σ). This leads to two sub-cases:

Case 2a: Suppose x is a non-world variable. Then, provided the Herbrand-Universe contains more than one element, we can substitute a ground term other than t, say t', for x. Since $\Sigma gr(L2) \neq \emptyset$, and since the Σ-consistency of L2 does not depend on the actual instance of non-world terms, L2{t'/x} must have at least one Σ-ground instance which is also a Σ-ground instance of L2. But since $t' \neq t$, it cannot be a ground instance of L1. Thus, $\Sigma gr(L1) \supseteq \Sigma gr(L2)$ cannot hold.

One might argue, if t' occurs in the path only, then t' does not necessarily occur in a Σ-ground instance, because by definition 6.5 only the last world of a path shows up in a Σ-ground instance. However, it follows from lemma 6.4, that no two paths end in the same final world, unless they are fully identical. Thus, if t occurs in one path and t' in the other, the corresponding Σ-ground instances cannot be identical. ■

Case 2b: Suppose x is a world variable. Thus, it occurs in the world path of L2. Let y be the immediate predecessor of x in the path, and instantiate x with f(y), where f( ) is the function used in the seriality axiom in Σ. Then, L2' = L2{f(y)/x} is still Σ-consistent, and therefore, by lemma 7.2, it possesses Σ-ground instances. Each of these is different from every Σ-ground instance of L1, because every instance of L1 has a term t, where L2' has a f( ). And $t \neq$ f( ), because f( ) does not occur in any path. It is not used in the initial translation, and by the same token, f( ) cannot be part of an MGU of any two paths. Therefore, no path during the course of a deduction can possibly contain f( ). Thus, Σgr(L1) $\supseteq$ Σgr(L2) cannot hold. Again, it does not make a difference, whether x occurs as the last element of the path or before. The same argument as in case 2a applies. ■■

What remains to be done to complete the above proof, is a proof of the following lemma:

Lemma 7.2 *Every Σ-consistent WPL literal has at least one Σ-ground instance.*

Proof: Theorem 6.4 states that the RML/CL literal, which corresponds to a Σ-consistent WPL literal, has a Σ-solvable constraint, if the WPL literal is Σ-consistent. And if its constraint is Σ-solvable, then the RML/CL literal has at least one Σ-ground instance. This follows from definition 5.1. Therefore, the RML/CL counterpart of the WPL literal has Σ-ground instances. But then, the WPL has Σ-ground instances too, since theorem 6.1 states that WPL literals have the same Σ-ground instances as their RML/CL counterparts. ■■

Summarizing this chapter, we have shown that subsumption of WPL unit clauses can be tested in the same way as for regular FOPL predicates, that is by searching for a substitution. The next section will establish a similar result for clauses with more than one literal.

B. MULTILITERAL CLAUSES

While unit clauses consist of exactly one literal, we use the term 'multiliteral clause' for clauses with an unrestricted number of literals, not necessarily more than one. In this sense, every clause is a multiliteral clause, even a unit clause. However, we find this terminology useful to allow for a clear distinction between clauses which necessarily have exactly one literal, and clauses that do not.

The main difference between subsumption in the two cases is that clauses can be of different length and yet subsume each other. For instance, the clause $(p \vee q)$ subsumes the clause $(p \vee q \vee r)$. As another difference, multiliteral clauses can be tautologies as in $(p \vee \neg p)$, which unit clauses cannot be. Although tautological clauses are subsumed by every other clause, if we take the definition of subsumption strictly, the task of detecting and deleting tautologies in a deduction system is usually considered a separate issue [Wos 93].

As in the case of unit clauses, we would like to treat a universally quantified WPL multiliteral clause as a scheme standing for its Σ-ground instances:

Definition 7.2 (Σ-ground Instances of Clauses, Set ΣGC) *Given a WPL clause $C = (L_1 \vee \ldots \vee L_n)$ and a substitution σ such that Cσ is variable free, LAST(Cσ) is a Σ-ground instance of C if and only if each $L_i\sigma$ is Σ-consistent, where $1 \leq i \leq n$.*
ΣGC(C) is a function that maps C to the set of all such Σ-ground instances of C.

Our motivation to require *all* literals in a ground clause to be Σ-consistent, as opposed to just one literal, is the goal to keep a WPL expression equivalent to its RML/CL counterpart. If the WPL clause is

$$C = (L_1 \vee \ldots \vee L_n) \quad (7.1)$$

then the corresponding RML/CL is

$$C' = (s_1/c_1 \vee \ldots \vee s_n/c_n) \quad (7.2)$$

where $s_i = LAST(L_i)$ and $c_i = constraint(path(L_i))$ for each i from 1 to n. (7.2) is equivalent to

$$C' = (c_1 \rightarrow s_1) \vee \ldots \vee (c_n \rightarrow s_n) \quad (7.3)$$

which in turn is equivalent to

$$C' = (s_1 \vee \ldots \vee s_n) / (c_1 \wedge \ldots \wedge c_n) \quad (7.4)$$

It follows from (7.4) and from the definition of Σ-ground instances (definition 5.1) that all constraints c_1, ..., c_n need to be Σ-solvable simultaneously. Thus, if we want C and C' to have the same Σ-ground instances, it is necessary to have the paths of all literals in C Σ-consistent, not just one.

Conjecture 7.1 (Ground Instance Equivalence) *Given a WPL clause*

$$C = (L_1 \vee \ldots \vee L_n)$$

and an RML Constraint Logic clause

$$C' = (s_1 \vee \ldots \vee s_n) \,/\, (c_1 \wedge \ldots \wedge c_n)$$

where $s_i = LAST(L_i)$ *and* $c_i = constraint(path(L_i))$ *for each i from 1 to n, then* $\Sigma GC(C) = \Sigma gr(C')$.

A proof of conjecture 7.1 would probably parallel the proof of theorem 6.1, which states a similar relationship about ground literals. Using our definition of WPL ground clauses, let us now return to the topic of subsumption.

As opposed to the unit clause case, "C1 subsumes C2" does not mean ΣGC(C1) ⊇ ΣGC(C2), because the Σ-ground clauses of C2 may be longer than the Σ-ground clauses of C1, and still be subsumed. Treating a clause as a set of its literals, we can however establish the following relationship:

Lemma 7.3 *Given two WPL clauses C1 and C2, C1 subsumes C2, if* $\forall C2'_{:C2' \in \Sigma GZ(C2)}\ \exists C1'_{:C1' \in \Sigma GZ(C1)}\ C1' \subseteq C2'$.

Proof: C1' entails C2', because all literals in the disjunction C1' occur also in the disjunction C2'. And if every Σ-ground clause of C2 is entailed by some Σ-ground clause of C1, then $\forall^*$ C1 ⊨ $\forall^*$ C2, because $\forall^*$ C1 in turn entails all of its Σ-ground clauses. But by definition 7.1, $\forall^*$ C1 ⊨ $\forall^*$ C2 means, C1 subsumes C2. ■■

The next theorem lifts this result to the level of variables.

Theorem 7.2 (Clausal Subsumption) *Given two WPL clauses C1 and C2, C1 subsumes C2, if there exists a substitution σ such that* $C1\sigma \subseteq C2$.

Proof: Pick any Σ-ground clause C2' of C2. Then there is a substitution μ such that C2' = C2μ. Consider C1' = C1$\sigma\mu$. All of its literals are also in C2', thus they are all Σ-consistent, because as a Σ-ground clause, all literals in C2' are Σ-consistent according to definition 7.2. But then C1' is also a Σ-ground clause. Hence, for every Σ-ground clause C2' of C2 there is a Σ-ground clause of C1' of C1 such that C1' ⊆ C2'. Thus, by lemma 7.3, C1 subsumes C2. ■■

Notice that theorem 7.2 uses "if" instead of "if and only if". In fact, the "only if" part does not work. For one, tautological clauses are entailed by every other clause. But even if we exclude tautological clauses from our treatment, there are cases where a clause C1 entails a clause C2 without the existence of a substitution σ such that C1σ ⊆ C2. The following example is taken from [Loveland 78]:

$$\forall x \;\; p(x) \rightarrow p(g(x)) \;\; \models \;\; \forall x \;\; p(x) \rightarrow p(g(g(x))) \qquad (7.5)$$

[21]

The clause on the left, call it C1, subsumes the clause on the right, C2, but no single instance of C1 is a subformula of C2. On the ground clause level, no ground clause of C1 implies any ground clause of C2. C1 and C2 do not have any ground clause in common. However, each ground clause of C2 is entailed by two ground clauses of C1. For instance, p(a)→p(g(g(a))) is entailed by p(a)→p(g(a)) and p(g(a))→p(g(g(a))) together.

We could avoid this problem, if we restricted the clauses such that a literal may not occur positively and negatively within the same clause. This restriction would solve the problem of tautological clauses as well. Thus, using this restriction, the "if" in theorem 7.2 could be replaced by an "if and only if". This approach is not practical though, because deductions cannot avoid dealing with clauses, in which the same predicate occurs twice. In fact, there are many cases where subsumption among clauses of this kind can be successfully detected using the substitution criteria. As an example, consider this slight variation of the clauses (7.5) [Loveland 78]:

$$\forall x \;\; p(x) \rightarrow p(g(y)) \;\; \models \;\; \forall x \;\; p(x) \rightarrow p(g(g(x))) \qquad (7.6)$$

Reflecting, what can and what cannot be detected using the substitution criteria, Loveland introduces a different definition [Loveland 78] of subsumption which we adopt for WPL:

---

21) These clauses are FOPL clauses, but since the language of WPL is a superset of the language of FOPL, they are also WPL clauses.

Definition 7.3 (θ-subsumption) *A WPL clause C θ-subsumes a WPL clause D iff there exists a substitution θ such that $C\theta \subseteq D$ and C has no more literals than D.*

Using this definition, (7.6) is a case of θ-subsumption, while (7.5) is not. And as far as tautological clauses, they are θ-subsumed by some other clause only if a part of the tautological clause is actually an instance of the subsuming clause. It is easy to see that every case of θ-subsumption is also a case of subsumption. This follows immediately from the definition and from theorem 7.2.

Definition 7.3 also takes care of another problem which we have not addressed before: a clause subsumes its factor. If two or more literals of a clause C have an MGU σ, then $C\sigma$ is called a *factor* of C. For instance, p(g(y)) is a factor of and is subsumed by (p(x) $\vee$ p(g(y))). This would appear to call for the deletion of the factor. But resolution is known to be incomplete without factoring. Definition 7.3's restriction that C may not have more literals than D makes sure that D is not a factor of C.

It turns out that the weaker θ-subsumption is a more useful deletion criteria than plain subsumption. The distinction between θ-subsumption and regular subsumption was not needed in the previous section, because there is no difference when only unit clauses are considered. Neither can a unit clause be a tautology, nor can a unit clause contain the same predicate in a positive literal and in a negative literal at the same time, nor can a unit clause be a factor of another unit clause.

In summary, this section extended the results for unit clauses to clauses with no restrictions on the number of variables. Subsumption can be tested in the same way as for regular FOPL clauses, that is by searching for a substitution. We defined θ-subsumption to account for the special problems mentioned above. None of these problems are due to the modal character of WPL, all of them are also prevalent in ordinary FOPL. An actual algorithm for θ-subsumption detection will be presented in the next section.

C. Algorithmic Subsumption Detection

The test for θ-subsumption is best being processed using the deduction machinery already available. Given two WPL clauses C and D, the method basically tries to refute $\forall^* C \wedge \neg\forall^* D$. Since the second conjunct is equivalent to $\exists^* \neg D$, skolemization requires the instantiation of the variables in C2 with distinct new constants that do not occur in either C or D. This instantiation can also be understood as protecting them against being substituted in the search of a substitution θ such that $C\theta \subseteq D$. In the special case where C and D are unit clauses, this process is sometimes referred to as 'half unification' [Wos 93], because we are looking for a unifier that affects only one side, namely C.

As an important result of the two preceding sections, subsumption in World Path Logic can be tested in the same way as in ordinary FOPL. In essence, it consists of the search for a substitution. Thus, any subsumption test for ordinary FOPL will also work for WPL. Table VI shows such a subsumption test algorithm. The algorithm employs resolution to find out, if such a substitution exists. As covered in Section IV.D, resolution in WPL is not significantly different from ordinary first order resolution. Resolution again involves unification, and it is only at that level where World Path Logic subsumption tests really differ from ordinary FOPL subsumption. Special purpose unification procedures are needed as described in Section IV.C.

As for the given subsumption test algorithm however, this does not make a difference, because it does not specify the particular details of unification. It just uses unification.

The algorithm in table VI is taken from [Chang, Lee 73] and [Robinson 65] with step 1 added to check for factorization. The following example illustrates the procedure. Consider the clauses

$$C = \neg P(0 \rightarrow sk_3 \rightarrow w_1, x) \vee Q(0 \rightarrow w_2, f(x), a)$$

and $$D = \neg P(0 \rightarrow w_3, z) \vee Q(0 \rightarrow sk_1 \rightarrow sk_2, f(h(y)), a) \vee \neg P(0 \rightarrow sk_3, h(y))$$

in a reflexive and transitive logic (KT4). Running the algorithm on C and D, it turns out that C θ-subsumes D. The algorithm's execution is traced in table VII.

Theorem 7.3 (Correctness of Algorithm) *Given two WPL clauses C and D, the algorithm listed in table VI terminates with "C θ-subsumes D" if and only if C θ-subsumes D. Furthermore, the algorithm is guaranteed to terminate.*

Table VI. Subsumption Test in WPL - An Algorithm

Input: Two WPL clauses, C and D $(D = D_1 \vee \ldots \vee D_m)$

(1)	*if # of literals in C > # of literals in D* *then output "factorization", stop*
(2)	*let* $\mu = \{ a_1/x_1, \ldots, a_n/x_n \}$, *where* $x_1 \ldots x_n$ *are the variables in D and* $a_1 \ldots a_n$ *are new constants, not occurring in C or D*
(3)	*set* $W = \{ \neg D_1\mu , \ldots , \neg D_m\mu \}$, *a set of unit clauses*
(4)	*set* $k = 0$
(5)	*set* $U^0 = \{C\}$, *a set of clauses*
(6)	*if* U^k *contains* ⊠ *then output " C* θ*-subsumes D ", stop*
(7)	*let* $U^{k+1} = \{$ *resolvents of all* C_1 *and* $C_2 \mid C_1 \in U^k$ *and* $C_2 \in W \}$
(8)	*if* $U^{k+1} = \emptyset$ *then output " C does not* θ*-subsume D ", stop* *else set* $k = k+1$, *goto (6).*

Table VII. Subsumption Test, An Example

Input:	$C = \neg P(0 \to sk_3 \to w_1, x) \vee Q(0 \to w_2, f(x), a)$ $D = \neg P(0 \to w_3, z) \vee Q(0 \to sk_1 \to sk_2, f(h(y)), a) \vee \neg P(0 \to sk_3, h(y))$
Step (2):	$\mu = \{ sk_4/w_3 , b/y , c/z \}$
Step (3):	$W = \{ P(0 \to sk_4, c) , \neg Q(0 \to sk_1 \to sk_2, f(h(b)), a) , P(0 \to sk_3, h(b)) \}$
Step (4):	$k = 0$
Step (5):	$U^0 = \{ \neg P(0 \to sk_3 \to w_1, x) \vee Q(0 \to w_2, f(x), a) \}$
Step (7):	$U^1 = \{ \neg P(0 \to sk_3 \to w_1, h(b)), Q(0 \to w_2, f(h(b)), a) \}$
Step (8):	$k = 1$
Step (7):	$U^2 = \{$ ⊠ $\}$
Step (8):	$k = 2$
Step (6):	" C θ-subsumes D "

Proof: "if": If C θ-subsumes D, then, by definition, C has no more literals than D. Thus, the algorithm does not terminate in step 1. Also by definition, there exists a θ such that $C\theta \subseteq D$. Let $D' = \{ D_1, \ldots, D_k \}$ be the literals in D such that $D' \subseteq D$ and $C\theta = D'$. Let $W' = \{ \neg D_1\mu , \ldots , \neg D_k\mu \}$, $W' \subseteq W$. Then there exists a linear ground refutation of $\{C\theta\mu\} \cup W'$, where $R_0 = C\theta\mu$, and $R_i = R_{i-1} - D_i\mu$, is a deduction sequence of ground clauses such that each R_i is an instance of a member of U_i. Clearly, R_k = ⊠, hence ⊠ $\in U_k$, thus the algorithm terminates in step 6 with " C θ-subsumes D ".

"<u>only if</u>": Suppose the algorithm terminates in step 6 with " C θ-subsumes D ". Then there is a linear deduction of the empty clause ⊠ (in U_k) with C as the top clause (in U_0), where each of the k resolution steps removes one literal of an instance of C. Let σ be the substitution comprised of all unifiers of this particular deduction. σ effects only C, since the literals in W are already ground. Thus, $C\sigma = D'\mu$ where D' is the partial clause of D consisting of those k literals that contributed to the deduction of ⊠. Consider the constants in μ. By their choice in step 2, they do not occur in C nor in D'. Replace all occurrences of these constants in σ by the variable, they are substituting in μ. Let the result of this operation be θ. Then $C\theta = D'$. Therefore, $C\theta \subseteq D$. Furthermore, the number of literals in C does not exceed the number of literals in D. Otherwise the algorithm would have stopped in step 1.

<u>Termination property</u>: Since all clauses in W are unit clauses, the resolvents in U^i have one literal less then their ancestors in U^{i-1}. Thus, U^k will eventually either be empty or contain the empty clause. ■■

Note that the algorithm correctly decides the clauses from example (7.5), where $C = \forall x\ p(x) \rightarrow p(g(x))$ and $D = \forall x\ p(x) \rightarrow p(g(g(x)))$, not to be a case of θ-subsumption. Although $C \wedge \neg D$ is unsatisfiable, the empty clause is not derived. Responsible for this is the resolution strategy which allows clause C to be used only once. And in fact, it would always take two ground instances of C to derive one ground instance of D. Thus, C subsumes but does not θ-subsume D.

The test for θ-subsumption can be quite expensive at times, as several unification and resolution operations are involved. On the other hand, subsumption tests can shorten the length of deductions drastically. [Loveland 78] gives an example in which forward and backward subsumption reduces the length of a refutation from some 100 clauses down to 12 clauses. So is subsumption worth the effort? If it is our primary goal to find a short proof, then forward and backward subsumption as well as the previously mentioned ancestor subsumption should be employed to its fullest extent. If, however, it is more important to find a proof fast, then one would probably be better off with a compromise of some sort. The possible gain by keeping the number of clauses small is paid for with the costs for subsumption tests, with decreasing returns when the literal count rises. Subsumption tests are most efficient when one of the clauses is a unit clause, and some implementations restrict its application to just unit or two-literal clauses. However, it is difficult to make a definite statement as to the optimal degree of using

θ−subsumption, since its benefits depend too much on implementational aspects and on not yet well enough researched problem qualities [Loveland 78].

D. SUMMARY

Recapitulating this chapter, the problem of subsumption in modal logic, when translated into clausal World Path Logic, parallels that of subsumption in ordinary clausal first order predicate logic. We have proven in Sections A and B that a WPL clause C subsumes a clause D, if there exists a substitution θ such that $C\theta \subseteq D$, just like in FOPL. Any subsumption test algorithm that works for FOPL will work for WPL as well. We have presented one possible algorithm. It relies on resolution to find the substitution. As we showed in Section IV.D, resolution in WPL works just like in FOPL. Resolution again is based on unification. So it is only at the level of unification where the special WPL needs make a difference.

This is kind of nice, we can basically upgrade any old FOPL theorem prover into a theorem prover for World Path Logic. The only change needed is the special purpose unification routine as outlined in Section IV.C. And if we add on another front-end translator from modal logic to WPL, we have a modal logic theorem prover at our hands. Once these changes have been taken care of, the whole other theorem proving machinery comes for free, including subsumption.

VIII. EXTENSIONS

To keep the presentation simple, the treatment in the previous chapter was based on several assumptions as to the modal logic under consideration. These restrictions were:

- we assumed the domain to be constant in all worlds
- the logics were implicitly monomodal, i.e. there was just one accessibility relation
- we admitted only those accessibility axioms that can be represented in definite clauses, i.e. reflexivity, symmetry, transitivity, euclidian. Furthermore, the accessibility relation was required to be serial

While these restrictions still leave us with a broad variety of modal systems, sufficient enough to cover many applications, it is worthwhile to explore what lies beyond. In this chapter we want to discuss the implications of dropping the assumptions above, and what needs to be done to extend our system to (a) varying domain logics, (b) multimodal logics, and (c) different accessibility axioms.

A. Varying Domain Logics

By maintaining world paths with the predicates, we made sure to resolve literals only within the scope of the same world, thus taking into account that predicates are subject to different true/false evaluations in different worlds. Similarly, the world path is kept track of in an additional argument to non-rigid functions (and constants) in order to account for world-dependent interpretations. Variables, however, had no world paths associated with them.

When the domain is not assumed to be the same in every world, then it becomes important which world a variable belongs to. A formula with variables is a scheme standing for the set of all its ground instances. But a variable from world i cannot be instantiated with just anything, it represents only the elements from world i's domain D_i. Thus, given a variable x from world i and a term t from world j, they unify only if t also exists in i. Now, when we assumed a constant domain, this was not a concern, because every element from D_j was then also an element of D_i, regardless of i and j.

But even when the domains are not given to be the same in all worlds, they are usually not completely unrelated. Their relationship is expressed in terms of the Barcan-Formula (BF) and its converse (FB). If the Barcan formula is a theorem of the logic, then the domains of all accessible worlds are subsets of the current world's domain. Conversely, if FB holds, then they are supersets of the current world's domain. Table VIII gives the two modal logic formulas and states their meaning in terms of accessibility and domains.

Table VIII. The Barcan Formula (BF) and its Converse (FB)

Name	Modal Logic formula	Relation between domains
BF	$\forall x \Box p(x) \rightarrow \Box \forall x\, p(x)$	if K(i,j) then $D_i \supseteq D_j$
FB	$\Box \forall x\, p(x) \rightarrow \forall x \Box p(x)$	if K(i,j) then $D_i \subseteq D_j$

In order to account for varying domains, we change the method as follows, combining ideas from [Cialdea 86] and [Jackson, Reichgelt 87]: First, the translation function from modal logic to WPL (table III) needs to be extended such that each occurrence of a variable is indexed with the path of the world, in which the variable was introduced. In particular, the line in table III that dealt with universal quantifiers is changed from:

- $t(s,X,\forall x\ \varphi) = t(s,X \cup \{x\},\varphi)$

to:

- $t(s,X,\forall x\ \varphi) = t(s,X \cup \{x_s\},\varphi\mu)$, where $\mu = \{x_s/x\}$

Next, the unification method needs to be upgraded. A term t (variable or non-variable) and a variable x can only be unified, if t exists in the domain of x. Let *path(x)* be the path associated with x, and let *path(t)* be the path of t, that is t's index if t is a variable, or t's first argument if t is a function term. Then we need to distinguish four cases, depending on which combination of the Barcan formulas holds:

(a) FB holds. Then the domains are monotonously increasing along the access path. x and t unify only if x's world is reachable from t's world. σ is a unifier of x and t, only if *path(t)σ* is a prefix of *path(x)σ*. We call this *prefix-unification*.

(b) BF holds. In this case, the domains are monotonously decreasing along the access path. Conversely to (a), x and t unify only if t's world is reachable from x's world. σ is a unifier of x and t, only if *path(x)σ* is a prefix of *path(t)σ*.

(c) Neither BF nor FB hold. Then there is no defined relation among the worlds' domains. A possible unifier σ of x and t has to comply to both restrictions of (a) and (b), i.e. *path(t)σ* is a prefix of *path(x)σ* and *path(x)σ* is a prefix of *path(t)σ*. In result, the paths have to be unifiable, such that *path(t)σ* = *path(x)σ*. In other words, x and t have to be associated with the same world.

(d) Both BF and FB hold. In this case, if world j is accessible from world i, then $D_i \subseteq D_j$ and $D_i \supseteq D_j$, thus $D_i = D_j$. This is a constant domain logic, and unification can be performed regardless of *path(x)* and *path(t)*.

Note that the special requirements of prefix unification come on top of the E-unification method for world paths as outlined in Section IV.C. The following example will illustrate WPL deduction in a varying domain logic.

Consider a logic, in which the WPL unification method employs prefix-unification to reflect the Barcan formula, as described in case (b) above. Our goal is to prove that the Barcan formula

$$\forall x\ \Box p(x) \rightarrow \Box\forall x\ p(x) \tag{8.1}$$

is actually a theorem in this logic. First, (8.1) needs to be negated, so we can do a refutation proof later on:

$$\forall x\ \Box p(x)\ \wedge\ \Diamond\exists x\ \neg p(x) \tag{8.2}$$

Translation into WPL, using the upgraded translation function, yields the clauses:

$$p(0 \rightarrow w, x_0) \tag{8.3}$$

$$\neg p(0 \rightarrow sk, f(0 \rightarrow sk)) \tag{8.4}$$

Resolution of (8.3) and (8.4) completes the refutation and yields the empty clause ⊠. The necessary unifier is $\sigma = \{sk/w,\ f(0 \rightarrow sk)/x_0\}$. Note that the latter substitution is allowed, since the index of x, 0, is a prefix of the path 0→sk.

If the logic under consideration is symmetric, then the Barcan formula implies its converse and vice versa. Suppose the Barcan formula is a theorem and K(i,j) holds for some i,j. Then by table VIII, $D_i \supseteq D_j$. But K(i,j) implies K(j,i) by symmetry. Thus, $D_j \supseteq D_i$. Hence $D_i = D_j$, and we have in effect a constant domain logic, which means that both BF and FB hold jointly.

We conjecture that deduction in varying domain World Path Logic, using the restricted unification method outlined above, is sound. The subsumption detection algorithm from Section VII.C should work as well, since it is based on resolution and unification.

B. Multimodal Logics

As mentioned in Section III.C, reasoning about the knowledge and belief of agents requires distinct modal operators, like $\Box_A$, $\Diamond_A$, $\Box_B$, $\Diamond_B$ and so forth, where the subscript indicates the agent. Thus, there are multiple accessibility relations, one for each agent.

The translation function from modal logic to WPL converts the modal operators into world paths. For instance, $\Box \Diamond P$ translates to $P(0 \rightarrow w \rightarrow sk)$. The arrow in the path can be viewed as a binary infix operator representing the accessibility relation. So $0 \rightarrow w \rightarrow sk$ is equivalent to $K(0,w) \wedge K(w,sk)$. Now when we have to deal with different accessibility relations, we also need to introduce distinct path infix operators. Two lines need to be changed in the definition of the translation function (table III). The monomodal version was:

- $t(s,X,\Box\varphi) = t(s \rightarrow w,X,\varphi)$
- $t(s,X,\Diamond\varphi) = t(s \rightarrow sk(X),X,\varphi)$

The new multimodal version is:

- $t(s,X,\Box_K\varphi) = t(s \rightarrow_K w,X,\varphi)$
- $t(s,X,\Diamond_K\varphi) = t(s \rightarrow_K sk(X),X,\varphi)$

where the subscript letter K indicates the agent. Now, $\Box_A \Diamond_B P$ is translated to the WPL formula $P(0 \rightarrow_A w \rightarrow_B sk)$.

As far as unification is concerned, paths unify only if the infix operators match. For instance, $P(0 \rightarrow_A sk)$ and $P(0 \rightarrow_B w)$ do not unify. However, $P(0 \rightarrow_A w_1)$ and $P(0 \rightarrow_B w_2)$ unify with

$\sigma = \{1/w_1, 1/w_2\}$, if the accessibility relations are reflexive. Recall the axioms for E-unification from Section IV.C:

- Reflexivity: $\forall w \quad w \to 1 \equiv w$
- Symmetry: $\forall w,w' \quad w \to w' \to w'^{-1} \equiv w$
- Transitivity: $\forall w,w',w'' \quad w \to w' \to w'' \equiv w \to (w' \to w'')$

Upgrading E-unification for the multimodal case, one set of axioms each is needed for every agent. In a two agent transitive logic KD4, for instance, the equational theory amounts to these two axioms:

$$\forall w,w',w'' \quad w \to_A w' \to_A w'' \equiv w \to_A (w' \to_A w'')$$

$$\forall w,w',w'' \quad w \to_B w' \to_B w'' \equiv w \to_B (w' \to_B w'')$$

Note that these axioms cannot be applied across different agents. In particular, the predicates $P(0 \to_A w)$ and $P(0 \to_A sk_1 \to_B sk_2)$ do not unify, because the paths $0 \to_A (sk_1 \to_B sk_2)$ and $0 \to_A sk_1 \to_B sk_2$ are not equal under the equational theory.

When Multimodal Logic is used to formalize reasoning about knowledge of agents, then usually the same accessibility restrictions hold for all agents. The question as to which modal logic to use, needs careful consideration. All the accessibility axioms listed in table I have their specific epistemic interpretation [Scherl 92]. The seriality schema D, $\Box A \to \Diamond A \equiv \neg\Box A \vee \neg\Box\neg A \equiv \neg(\Box A \wedge \Box\neg A) \equiv \neg\Box(A \wedge \neg A) \equiv$

$$\neg\Box(\text{false})$$

can be interpreted as saying that the agent's belief is consistent. The reflexivity schema T

$$\Box A \to A$$

states that everything that is known is true, while the transitivity axiom 4

$$\Box A \to \Box\Box A$$

states that if an agent knows something, he knows that he knows it.

Let us now do an actual example of a proof in multimodal logic, and return to the Two Wise-Men puzzle from Section III.C.2. The problem was formalized in the modal logic sentences (3.1)-(3.4), which are reprinted here as (8.5) through (8.8). For this particular example, the accessibility restrictions are of no concern, since it turns out that the E-unification equality axioms are not needed for the proof.

$$\Box_A (\neg spot(A) \to \Box_B \neg spot(A)) \qquad (8.5)$$

$$\Box_A \Box_B(\mathrm{spot}(A) \vee \mathrm{spot}(B)) \quad (8.6)$$

$$\Box_A \neg \Box_B \mathrm{spot}(B) \quad (8.7)$$

The hypothesis to be proven is:

$$\Box_A \mathrm{spot}(A) \quad (8.8)$$

Negation of the hypothesis and conversion of the sentences into negation normal form yields:

$$\Box_A (\neg \mathrm{spot}(A) \rightarrow \Box_B \neg \mathrm{spot}(A)) \quad (8.9)$$

$$\Box_A \Box_B(\mathrm{spot}(A) \vee \mathrm{spot}(B)) \quad (8.10)$$

$$\Box_A \Diamond_B \neg \mathrm{spot}(B) \quad (8.11)$$

$$\Diamond_A \neg \mathrm{spot}(A) \quad (8.12)$$

The next step is translation into WPL:

$$\mathrm{spot}(0 \rightarrow_A w_1, A) \vee \neg \mathrm{spot}(0 \rightarrow_A w_1 \rightarrow_B w_2, A) \quad (8.13)$$

$$\mathrm{spot}(0 \rightarrow_A w_3 \rightarrow_B w_4, A) \vee \mathrm{spot}(0 \rightarrow_A w_3 \rightarrow_B w_4, B) \quad (8.14)$$

$$\neg \mathrm{spot}(0 \rightarrow_A w_5 \rightarrow_B sk_1, B) \quad (8.15)$$

$$\neg \mathrm{spot}(0 \rightarrow_A sk_2, A) \quad (8.16)$$

The deduction sequence is as follows:

$$\neg \mathrm{spot}(0 \rightarrow_A sk_2 \rightarrow_B w_2, A) \qquad \text{[resolvent of 8.13, 8.16]} \quad (8.17)$$

$$\mathrm{spot}(0 \rightarrow_A w_3 \rightarrow_B sk_1, A) \qquad \text{[resolvent of 8.14, 8.15]} \quad (8.18)$$

$$\boxtimes \qquad \text{[resolvent of 8.17, 8.18]} \quad (8.19)$$

Thus, the wise-man A knows, that he has a white spot on his forehead.

[Scherl 92] proved this deduction method for multimodal logic to be sound and complete. We conjecture that subsumption works, just as usual, by finding a substitution. The algorithm described in Section VII.C should do the job without changes, since all the special requirements for multimodal logics are hidden in the unification process.

C. Other Accessibility Restrictions

Throughout the previous chapters the accessibility relation was assumed to be serial. Also, as to the accessibility restrictions, we admitted only a subset of reflexivity, symmetry, transitivity, and the euclidian property.

Lifting any of these limitations has serious consequences as for the proofs in Chapters IV through VII. For instance, seriality ensures that a Σ-consistent path is also Σ-solvable. Consider resolving the empty clause ⊠ from the clauses $P(0 \rightarrow w_1 \rightarrow w_2)$ and $\neg P(0 \rightarrow w_3 \rightarrow sk_1)$. If we are not guaranteed that some world is reachable from world 0, then the deduction of ⊠ is not sound, because there is no corresponding ground refutation.

Imposing the accessibility restriction to be some combination of reflexivity, symmetry, and transitivity made sure that all clauses in the constraint theory Σ were definite clauses, i.e. clauses with exactly one positive literal. Suppose we specify the accessibility relation to be connected. That means, if both b and c are accessible from a, then either is b accessible from c or c from a. More formally, we can express connectivity as:

$$\forall w_1, w_2, w_3 \;\; K(w_1, w_2) \wedge K(w_1, w_3) \rightarrow K(w_2, w_3) \vee K(w_3, w_2) \tag{8.20}$$

Now consider the following set of WPL sentences:

$$\neg P(0 \rightarrow sk_1) \tag{8.21}$$

$$\neg Q(0 \rightarrow sk_2) \tag{8.22}$$

$$P(0 \rightarrow sk_2 \rightarrow w_1) \tag{8.23}$$

$$Q(0 \rightarrow sk_1 \rightarrow w_2) \tag{8.24}$$

These sentences are in fact unsatisfiable. Since both sk_1 and sk_2 are accessible from 0, we can either instantiate w_1 with sk_1, thus ⊠ is the resolvent of (8.21) and (8.23), or we can instantiate w_2 to sk_2, which would allow us to infer ⊠ from (8.22) and (8.24). None of the resulting paths $0 \rightarrow sk_2 \rightarrow sk_1$ and $0 \rightarrow sk_1 \rightarrow sk_2$ is Σ-consistent however, since neither $K(0,sk_2) \wedge K(sk_2,sk_1)$ nor $K(0,sk_1) \wedge K(sk_1,sk_2)$ can be inferred from Σ.

The problem of how to handle subsumption, when the accessibility relation is not serial, or when an accessibility restriction cannot be expressed in a definite clause, remains an unsolved question.

IX. CONCLUSION

Subsumption is a technique to detect redundancies among the sentences in the search space of automated deduction systems. Naturally, the way subsumption is done depends on the logic used in the deduction system. This dependency is of particular relevance when deduction in modal logic is concerned, because modern modal logic deduction methods do not perform the deduction directly in modal logic. Instead, they translate the modal logic expressions into some other target language, and then determine whether there exists a proof in that language.

Drawing from existing work, we defined and introduced World Path Logic (WPL) as such a kind of target language in Chapter IV. All these languages have in common that the possible worlds semantic of modal logic is made explicit in world access paths, which are kept as syntactical items with the predicates and terms of the language. Our translation function from modal logic to WPL differs from existing work in two points: skolemization is integrated into the translation procedure, and simplified world path structures are used. Deduction in World Path Logic is very similar to deduction in ordinary first order logic. The significant difference lies in a special purpose unification method for world paths. World paths are unified under an equational theory which represents the restrictions imposed on the accessibility relation. Thus, the accessibility restrictions are encoded into the unification algorithm. The method is somewhat restricted as to the properties of accessibility relation: it has to be serial and can be closed under any combination of reflexivity, symmetry, and transitivity. Thus, the modal logic systems KD, KT, KDB, KD4, KTB, KT4(S4), and KT5(S5) are supported. In contrast to ordinary unification, the most general unifier is unfortunately not always unique. Thus, in a resolution based system, multiple resolvents may need to be created.

We developed a deeper understanding of World Path Logic by relating it to another language, RML/CL, a first order predicate logic with restricted quantification. The results from Chapters V and VI justified why modal deduction works in World Path Logic, and provided us with the theoretical background for the following treatment of subsumption.

Using a standard definition of subsumption, we approached the topic by considering a WPL sentence as a scheme standing for the conjunction of its Σ-ground instances. This is not as trivial as it may seem, since not every possible variable free instance of S is considered 'legal'.

In result, however, the problem of subsumption in modal logic, when translated into clausal World Path Logic, parallels that of subsumption in ordinary clausal first order predicate logic. We were able to prove that a WPL clause A subsumes a clause B, if there exists a substitution θ such that $A\theta \subseteq B$, just like in FOPL. Any subsumption test algorithm, that works for FOPL, works for WPL as well. We have presented one possible algorithm. It relies on resolution to find the substitution. As we showed in Section IV.D, resolution in WPL works just like in FOPL. Resolution again is based on unification. So it is only at the level of unification where the special WPL needs make a difference.

Although the results have only been proven for certain modal logics, we conjecture based on the discussion in Chapter VIII, that the results can be extended to multimodal logics as well as to varying domain modal logics. That is, the necessary changes to account for these extensions effect only the special purpose unification method. Resolution and the subsumption test should still work like before.

We were not able to extend the method to non-serial logics and to logics where the accessibility relation restrictions cannot be axiomatized in definite clauses. Future work might try to work on these extensions, and to prove the extensions which we were only able to conjecture. Also, it can be worthwhile to explore subsumption outside the realm of those deduction methods that work by translation.

And of course, testing out the deduction and subsumption methods in an actual implementation should provide valuable insights. Actually, the implementation should not be too difficult. At least one does not need to build an automatic theorem prover from scratch. According to our results, we can basically upgrade any old FOPL theorem prover into a theorem prover for World Path Logic. The only change needed is the special purpose unification routine as outlined in Section IV.C. And if we add on a simple translator from modal logic to WPL, we have a modal logic theorem prover at our hands. Once these changes have been taken care of, all the other theorem proving machinery comes for free, including subsumption.

BIBLIOGRAPHY

[Abadi, Manna 86] Abadi, Martin and Manna, Zohar, "Modal theorem proving," in *Proceedings of the 8th International Conference on Automated Deduction*, Lecture Notes in Computer Science, vol. 230, ed. Jörg H. Siekmann, pp. 172-189, Springer-Verlag, Berlin, 1986

[Abadi, Manna 90] Abadi, Martin and Manna, Zohar, "Nonclausal Deduction in First-Order Temporal Logic," *Journal of the ACM*, vol. 37(2), pp. 279-317, 1990.

[Auffray, Enjalbert 89] Auffray, Yves and Enjalbert, Patrice, "Modal theorem proving: an equational viewpoint," in *Proceedings, International Joint Conference on Artificial Intelligence*, pp. 441-445, Morgan Kaufmann, 1989.

[Chang, Lee 73] Chang, Chin Liang and Lee, Richard Char-Tung, *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, Orlando, 1973.

[Cialdea 86] Cialdea, Marta, *Uné methode de déduction automatique en logique modale*, thesis, Université Paul Sabatier, Toulouse, 1986. (cited in [Cialdea 91])

[Cialdea 91] Cialdea, Marta, "Resolution for some first-order modal systems," *Theoretical Computer Science*, vol. 85, pp. 213-229, 1991.

[Fariñas 82] Fariñas Del Cerro, Luis, "A simple deduction method for modal logic," in *Information Processing Letters*, vol. 14(2), pp. 49-51, April 20, 1982.

[Fariñas 91] Fariñas del Cerro, Luis and Herzig, Andreas, "Modal deduction with applications in epistemic and temporal logics," to appear in *Handbook of Logic in AI*, ed. A. Galton, D. Gabbay, Ch. Hogger, Oxford University Press 366/92, 5th draft, July 1991.

[Fitting 83] Fitting, Melvin, *Proof Methods for Modal and Intuitionistic Logics*, D. Reidel, Dordrecht, Holland, 1983. (cited in [Wallen 90])

[Frisch, Scherl 91] Frisch, Alan M. and Scherl, Richard Brian, "A general framework for modal deduction," in *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, ed. J. A. Allen, R. Fikes, and E. Sandewall, Morgan Kaufmann, 1991.

[Geissler, Konolige 86] Geissler, Christophe and Konolige, Kurt, "A resolution method for quantified modal logics of knowledge and belief," in *Proc. of the Conf. on Theoretical Aspects of Reasoning about Knowledge*, ed. J. Halpern, pp 309-324, Morgan Kaufmann, 1986.

[Genesereth, Nilsson 87] Genesereth, Michael R. and Nilsson, Nils J., *Logical Foundations of Artificial Intelligence*, Morgan Kaufmann. Los Altos, California, 1987. (cited in [Scherl 92])

[Gent 92] Gent, Ian P., *Analytic Proof Systems for Classical and Modal Logics of Restricted Quantification*, PhD thesis, University of Warwick, Coventry, UK, March 1992.

[Gödel 31] Gödel, Kurt, "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I," *Monatshefte für Mathematik und Physik*, vol. 38, pp. 173-198, 1931. Translation in: Van Heijenoort, Jean, *From Frege to Gödel - A Source Book in Mathematical Logic 1879-1931*, pp 596-616, Harvard University Press, 1967.

[Herbrand 30] Herbrand, Jacques, *Recherches sur la théorie de la démonstration*, doctoral thesis, Chapter 5, Sorbonne University of Paris, June 1930. Translation in: Van Heijenoort, Jean, *From Frege to Gödel - A Source Book in Mathematical Logic 1879-1931*, pp 529-581, Harvard University Press, 1967.

[Jackson, Reichgelt 87] Jackson, Peter and Reichgelt, Han, "A general proof method for first order modal logic" in: *Proceedings, International Joint Conference on Artificial Intelligence*, pp. 942-944, Morgan Kaufmann, 1987

[Jackson, Reichgelt 89] Jackson, Peter and Reichgelt, Han, "A general proof method for modal predicate logic," in *Logic-based Knowledge Representation*, ed. P. Jackson, H. Reichgelt, and F. van Harmelen, pp. 177-228, Chapter 8, MIT Press, 1989.

[Kripke 63] Kripke, Saul A., "Semantical considerations on modal logic," *Acta Philosophica Fennica*, vol. 16, pp. 83-94, 1963. (cited in [Scherl 92])

[Loveland 78] Loveland, Donald W., *Automated Theorem Proving: A Logical Basis*, North-Holland Publishing Co., Amsterdam, 1978.

[Morgan 76] Morgan, Charles G. "Methods for automated theorem proving in nonclassical logics," *IEEE Transactions on Computers*, vol C-25, no. 8, pp. 852-862, August 1976.

[Ohlbach 88] Ohlbach, Hans Jürgen, "A resolution calculus for modal logics," in *9th International Conference on Automated Deduction*, ed. Ewing Lusk and Ross Overbeck, pp. 500-516, Springer-Verlag, Berlin, 1988.

[Ohlbach 93] Ohlbach. Hans Jürgen, "Translation Methods for Non-Classical Logics - An Overview," in *Working Notes, AAAI Fall Symposion Series*, pp. 113-125, October 22-24, 1993.

[Owicki, Lamport 82] Owicki, Susan and Lamport, Leslie, "Proving Liveness Properties of Concurrent Programs," in *ACM Transactions on Programming Languages and Systems*, vol. 4(3), pp. 455-495, July 1982.

[Pelletier 90] Pelletier, Francis Jeffrey, "Automated modal logic theorem proving in THINKER," 1990. (cited in [Scherl 92])

[Robinson 65] Robinson, J.A. "A machine oriented logic based on the resolution principle," *Journal of the ACM*, vol. 12, pp. 23-41, 1965.

[Scherl 92] Scherl, Richard Brian, *A Constraint Logic Approach to Automated Modal Deduction*, PhD thesis, University of Illinois at Urbana-Champaign, 1992.

[Schollmeyer, McMillin 93] Schollmeyer, Martina and McMillin, Bruce M., *Using Temporal Subsumption for Developing Efficient Error-Detecting Distributed Algorithms*, technical report no. CSC93-28, University of Missouri-Rolla, November 1993.

[Skolem 20] Skolem, Thoralf "Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit und Beweisbarkeit mathematischer Sätze nebst einem Theorem über dichte Mengen," *Videnskapsselskapets Skrifter, I. Mat. Naturv. Klasse*, no. 4, pp. 1-36, Oslo, September 1920

[Styazhkin 69] *History of Mathematical Logic from Leibniz to Peano*, MIT Press, Cambridge, 1969.

[Wallen 90] Wallen, Lincoln A., *Automated proof search in non-classical logics*, MIT Press, Cambridge, Massachusetts, 1990.

[Wos 93] Wos, Larry, personal communication, September 1993.

VITA

Dirk Heydtmann was born on May 19, 1964, in Schleswig, Germany. Majoring in Mathematics and Physics, he received an Abitur degree from Domschule Schleswig in 1983. Subsequently, he joined a three year vocational training program with a local bank. Starting in 1986, he studied Computer Science and Business Administration at Fachhochschule Flensburg College, Germany, and was awarded a Dipl-FH degree in 1990.

Following graduation, he worked for one and a half years as a systems analyst for Krupp MaK Data System, a consulting firm out of Kiel, Germany. He left this engagement in July 1991, when he was awarded a Fulbright Scholarship for graduate studies in the USA. Since then, he has been a graduate student with the University of Missouri-Rolla, where he is currently completing a Master's degree in Computer Science.