MAX-PLANCK-INSTITUT
FÜR
INFORMATIK

First-Order Modal Logic Theorem Proving and Standard PROLOG

Andreas Nonnengart

MPI–I–92–228 July 1992

INFORMATIK

Im Stadtwald
W 6600 Saarbrücken
Germany

Author's Address

Andreas Nonnengart
Max-Planck-Institut für Informatik
Im Stadtwald
W-6600 Saarbrücken 11
Germany
nonnenga@mpi-sb.mpg.de

Acknowledgements

I want to thank Detlef Fehrer, Ullrich Hustadt, Hans Jürgen Ohlbach, Renate Schmidt and Christoph Weidenbach for the many fruitful discussions, helpful remarks and careful reading of earlier versions.
This work has been supported by the Bundesministerium für Forschung und Technologie (BMFT), Project LOGO.

Abstract

Many attempts have been started to combine logic programming and modal logics. Most of them however, do not use classical PROLOG, but extend the PROLOG idea in order to cope with modal logic formulae directly. These approaches have the disadvantage that for each logic new logic programming systems are to be developed and the knowledge and experience gathered from PROLOG can hardly be utilized.
Modal logics based on Kripke-style relational semantics, however, allow a direct translation from modal logic into first-order predicate logic by a straightforward translation of the given relational semantics. Unfortunately such a translation turns out to be rather naïve as the size of formulae increases exponentially during the translation.
This paper now introduces a translation method which avoids such a representational overhead. Its basic idea relies on the fact that any binary relation can be replaced by equations and inequations which (under certain circumstances) can be eliminated later on by some further transformation. The overall approach thus works essentially for any modal logic having a Kripke-style possible world semantics and first-order describable frame properties. If at all, its application as a pre-processing for PROLOG is limited merely by the possibility of having frame properties which are not Horn or not even first-order describable.

Keywords

Modal Logics, Logic Programming, PROLOG, Theorem Proving, Automated Deduction

Chapter 1

Introduction

Modal logics have become more and more interesting in the area of artificial intelligence in general and automated theorem proving in particular. Most attempts dealing with Knowledge and Belief, Time, Action, and Obligation are very closely related to modal logics. This forced logicians to, firstly, invent logics which are able to cope with the user's requirements and, secondly, to develop calculi which allow an automated reasoning within these logics.
As it is obvious that the utilization of standard Hilbert Calculi is by far not efficient enough, other calculi well-known from classical first-order theorem proving have been taken as a model for modal logics. For instance, the tableau method had been extended by Melvin Fitting (see [4]) in order to be able to deal with modal logics as well. Other applications of the tableau method to modal logics can be found in [5].
First attempts in which it was tried to allow the resolution method for reasoning in modal logics can be found for example in [1]. The main idea behind this method is to manipulate modal logic formulae by some set of transformation rules such that classical resolution is possible inside modal contexts.
Translation of modal logic formulae into first-order predicate logic goes back at least to Moore (see [6]). More elaborated techniques can be found in [3], [7] and [8].
The method proposed in this paper is in fact very closely related to these translation approaches. It overcomes most of the problems one gets by Moore's approach, as it is much more efficient. In the other translation approaches the respective modal logic properties are hidden behind some extra equational theory which can, in principle, be put into special theory unification algorithms. I.e. they require either a strong equality handling of the inference system used or even the possibility to deal with theory unification.
Standard PROLOG does not directly support this. Therefore, a method is proposed in this paper which allows derivations in the sense of [3], [7] and [8], and that without some extra mechanisms like equality handling or theory resolution. In fact, it even allows standard PROLOG to be utilized as a modal logic theorem prover since it overcomes the problem of infinite loops which would occur after the application

of Moore's attempt.

Most parts of this introductory chapter repeat the formal basis of modal logics. The main reason for this reminder is that the following chapters partly depend on the notation of the preliminaries. In order to avoid confusion with different kinds of notations used in the standard literature this repetition seems convenient. The reader interested in more detailed information on modal logic is referred to [5] and [2].

Definition 1.1 (Algebras and Structures)
As usual, an algebra is a pair consisting of a domain and a set of functions over this domain.
A structure is an algebra together with a set of relations over the algebra's domain.

Definition 1.2 (The Signature of Modal Logic)
Assume the following sets of symbols:

- $\mathbf{V}$ is a set of variable symbols
- $\mathbf{F}$ is a set of function symbols
- $\mathbf{P}$ is a set of predicate symbols

These are called the *non-logical symbols* of the modal logic language under consideration. The *logical symbols* are the well-known logical connectives and quantifiers together with the modal operators $\Box$ and $\Diamond$. The tuple $\Sigma_{\mathrm{M}} := (\mathbf{V}, \mathbf{F}, \mathbf{P})$ is then called a *modal logic signature.*

Once we know the alphabet of the language, we are able to build sentences, i.e. formulae, constructed from the given atoms and the logical connectives. The following definition provides this in more detail:

Definition 1.3 (Terms, Atoms, and Formulae)
Let Σ_{M} be a modal logic signature.
Terms, atoms and formulae are defined as follows:

- Each variable symbol is a term.
- If f is an n-place function symbol and $t_1, \ldots, t_n$ are terms then $f(t_1, \ldots, t_n)$ is a term.
- If P is an n-place predicate symbol and $t_1, \ldots, t_n$ are terms then $P(t_1, \ldots, t_n)$ is an atom.
- Each atom is a formula.
- If Φ and Ψ are formulae and x is a variable symbol then $\neg\Phi$, $\Phi \land \Psi$, $\Phi \lor \Psi$, $\Phi \Rightarrow \Psi$, $\Phi \Leftrightarrow \Psi$, $\forall x\Phi$ and $\exists x\Phi$ are formulae.
- If Φ is a formula then $\Box\Phi$ and $\Diamond\Phi$ are formulae.

Formulae are supposed to be either true or false. Their actual truth value depends on the way the respective symbols are interpreted. It is thus necessary to define what is understood by an interpretation:

Definition 1.4 (Frames and Interpretations)
By a frame $\mathcal{F}_{\mathrm{M}}$ we understand any pair $(\mathcal{W}, \Re)$ where $\mathcal{W}$ is a non-empty set (of worlds) and $\Re$ is an arbitrary binary relation on $\mathcal{W}$ called the *accessibility relation* between worlds.
By a Σ_{M}-interpretation $\Im_{\mathrm{M}}$ based on a frame $\mathcal{F}_{\mathrm{M}} = (\mathcal{W}, \Re)$ we understand any tuple $(\mathcal{D}, \mathcal{F}_{\mathrm{M}}, \Im_{\mathrm{loc}}, \tau, \phi)$ where

- $\mathcal{D}$ denotes a set of individuals; the universe of discourse
- $\mathcal{F}_{\mathrm{M}}$ is a frame
- $\Im_{\mathrm{loc}}$ is a mapping from $\mathcal{W}$ to the set of Σ_{M}-structures, where the respective domains all are identical to $\mathcal{D}$.
- τ denotes the actual world (the current situation)
- ϕ is a variable assignment, i.e. a function which maps domain variables to elements of the domain $\mathcal{D}$.

We will usually call $\Im_{\mathrm{M}}$ a *modal logic interpretation over* Σ_{M}.

Note that we assume a global universe of discourse, i.e. each element of the domain is known in any world. We thus consider a *constant domain* structure. Varying domains will be handled later.

Once it has been defined how interpretations look like, we are able to define how formulae are to be interpreted by such an interpretation. To this end a satisfiability relation between interpretations and formulae has to be provided such that a formula Φ is true with respect to the interpretation $\Im$ iff the pair $(\Im, \Phi)$ is an element of the satisfiability relation. As usual we begin with the interpretation of terms:

Definition 1.5 (Interpretation of Terms)
Let $\Im_{\mathrm{M}} = (\mathcal{D}, \mathcal{F}_{\mathrm{M}}, \Im_{\mathrm{loc}}, \tau, \phi)$ be an interpretation and let t be an arbitrary term.

$$\Im_{\mathrm{M}}(t) = \begin{cases} \phi(t) & \text{if } t \text{ is a variable symbol} \\ \hat{f}(\Im_{\mathrm{M}}(t_1), \ldots, \Im_{\mathrm{M}}(t_n)) & \text{if } t = f(t_1, \ldots, t_n) \end{cases}$$

where $\hat{f}$ is the interpretation of the symbol f in the structure $\Im_{\mathrm{loc}}(\tau)$. In the sequel we will use the simplified notation $(\Im_{\mathrm{loc}}(\tau))(f)$ instead (and similarly $(\Im_{\mathrm{loc}}(\tau))(P)$ for the interpretation of the predicate symbol P in the structure $\Im_{\mathrm{loc}}(\tau)$).

The following notation will be convenient for the sequel:

Definition 1.6
Let ϕ be a variable assignment. We define:

$$\phi(y)[x/a] = \begin{cases} a & \text{if } y = x \\ \phi(y) & \text{otherwise} \end{cases}$$

If $\Im_\mathrm{M} = (\mathcal{D}, \mathcal{F}_\mathrm{M}, \Im_\mathrm{loc}, \tau, \phi)$ then we usually abbreviate $(\mathcal{D}, \mathcal{F}_\mathrm{M}, \Im_\mathrm{loc}, \tau, \phi[x/a])$ by $\Im_\mathrm{M}[x/a]$ and $(\mathcal{D}, \mathcal{F}_\mathrm{M}, \Im_\mathrm{loc}, \chi, \phi)$ by $\Im_\mathrm{M}[\chi]$.

Now we are in a position where we can define the satisfiability relation $\models_\mathrm{M}$ as a relation between interpretations and formulae:

Definition 1.7 (Satisfiability)
Let $\Im_\mathrm{M} = (\mathcal{D}, \mathcal{F}_\mathrm{M}, \Im_\mathrm{loc}, \tau, \phi)$ be a Σ_M-interpretation where $\mathcal{F}_\mathrm{M} = (\mathcal{W}, \Re)$ is a frame. A formula Φ is said to hold for the interpretation $\Im_\mathrm{M}$ if and only if $\Im_\mathrm{M} \models_\mathrm{M} \Phi$ holds, where $\models_\mathrm{M}$ is recursively defined as follows:

$\Im_\mathrm{M} \models_\mathrm{M} P(\ldots, t_i, \ldots)$ iff $\Im_\mathrm{loc}(\tau)(P)(\ldots, \Im_\mathrm{M}(t_i), \ldots)$

The cases for the classical logical connectives should be clear

$\Im_\mathrm{M} \models_\mathrm{M} \forall x \Phi$	iff	$\Im_\mathrm{M}[x/a] \models_\mathrm{M} \Phi$ for every $a \in \mathcal{D}$
$\Im_\mathrm{M} \models_\mathrm{M} \exists x \Phi$	iff	$\Im_\mathrm{M}[x/a] \models_\mathrm{M} \Phi$ for some $a \in \mathcal{D}$
$\Im_\mathrm{M} \models_\mathrm{M} \Box \Phi$	iff	$\Im_\mathrm{M}[\tau'] \models_\mathrm{M} \Phi$ for every $\tau' \in \mathcal{W}$ such that $\Re(\tau, \tau')$
$\Im_\mathrm{M} \models_\mathrm{M} \Diamond \Phi$	iff	$\Im_\mathrm{M}[\tau'] \models_\mathrm{M} \Phi$ for some $\tau' \in \mathcal{W}$ with $\Re(\tau, \tau')$

An interpretation $\Im_\mathrm{M}$ is said to *satisfy* a formula Φ if $\Im_\mathrm{M} \models_\mathrm{M} \Phi$. Φ is called *satisfiable* then and the corresponding interpretation is called a *model* for Φ. As usual we call Φ *unsatisfiable* if no model for Φ exists.

The (propositional) modal logic K axiomatized by the axiom schema

$$\Box(\Phi \Rightarrow \Psi) \Rightarrow (\Box\Phi \Rightarrow \Box\Psi)$$

and the inference rules

$$\frac{\Phi}{\Box\Phi} \qquad \frac{\Phi, \Phi \Rightarrow \Psi}{\Psi}$$

contains exactly the theorems according to (the propositional fragment of) Definition 1.7.
Other modal logics are usually obtained by adding further axiom schemata as for example $\Box\Phi \Rightarrow \Phi$ or $\Box\Phi \Rightarrow \Box\Box\Phi$. It was Saul Kripke who found out that many interesting axiom schemata correspond to certain properties of the accessibility relation in the modal logic semantics. Some of these correspondences are listed below:

Axiom Schema	Property
$\Box\Phi \Rightarrow \Diamond\Phi$	$\forall x \exists y\, \Re(x,y)$
$\Box\Phi \Rightarrow \Phi$	$\forall x\, \Re(x,x)$
$\Phi \Rightarrow \Box\Diamond\Phi$	$\forall x,y\, \Re(x,y) \Rightarrow \Re(y,x)$
$\Box\Phi \Rightarrow \Box\Box\Phi$	$\forall x,y,z\, \Re(x,y) \wedge \Re(y,z) \Rightarrow \Re(x,z)$
$\Diamond\Phi \Rightarrow \Box\Diamond\Phi$	$\forall x,y,z\, \Re(x,y) \wedge \Re(x,z) \Rightarrow \Re(y,z)$

Historically these properties are called D, T, B, 4, and E respectively. Other modal logics than K can now be generated by adding an arbitrary sequence of these extra-properties. So e.g. the well-known modal logic S5 is referred to by KTB4 or equivalently by KTE. The modal logic S4 is simply KT4 and the logic M is KT.

What is remarkable for the properties from above is that they all are first-order properties. Therefore it is not too surprising that a translation of modal logic formulae into first-order predicate logic can easily be done for any of the modal logics just mentioned. However, such a translation cannot seriously be recommended for theorem proving purposes since it produces formulae which are considerably bigger than the original ones just because of the fact that so many additional literals have to be introduced which stem from the underlying accessibility relation.

Nevertheless we will provide its definition, for this can be used further on for a better and more efficient translation approach.

Chapter 2

Relational Translation

The definition of modal logic required the determination of signatures, terms and formulae, and interpretations. A translation from one logic into another therefore needs to translate each of these parts. We call such a translation a *logic morphism* divided into three parts: a signature morphism, a formula morphism, and an interpretation morphism.
The particular logic morphism defined in this chapter will be called Π and its respective parts Π_Σ, Π_F and $\Pi_\Im$.

Definition 2.1 (The Signature Morphism Π_Σ)
Let $\Sigma_\mathrm{M} := (\mathbf{V}, \mathbf{F}, \mathbf{P})$ be a modal logic signature.
For each f in $\mathbf{F}$ let f' be a new function symbol and for each P in $\mathbf{P}$ let P' be a new predicate symbol. Additionally we assume a sort symbol, W, which is supposed to represent the sort of worlds under consideration and D as a sort symbol representing the sort of individuals.
Then let $\mathbf{F}' = \{f' \mid f \in \mathbf{F}\} \cup \{\iota\}$, $\mathbf{P}' = \{P' \mid P \in \mathbf{P}\} \cup \{R\}$, and $\mathbf{S} = \{D, W\}$.
We then define: $\Pi_\Sigma(\Sigma_\mathrm{M}) = (\mathbf{V}, \mathbf{F}', \mathbf{P}', \mathbf{S})$
$\Pi_\Sigma(\Sigma_\mathrm{M})$ will usually be called the *(sorted) predicate logic signature generated from* Σ_M.

Obviously the symbols R and ι are supposed to represent the accessibility relation $\Re$ and the "actual" world τ respectively.

Definition 2.2 (The Formula Morphism Π_F)
There are actually two parts to be defined: a translation of terms *and* a translation of formulae. For convenience both are given by Π_F.
Let t be an arbitrary term and let u be a world term, i.e. either a variable symbol or the world constant ι. We define $\Pi_\mathrm{F}(t, u)$ by induction on the structure of t:

$$\begin{array}{lcl} \Pi_\mathrm{F}(x, u) & = & x \\ \Pi_\mathrm{F}(f(t_1, \ldots, t_n), u) & = & f'(u, \Pi_\mathrm{F}(t_1, u), \ldots, \Pi_\mathrm{F}(t_n, u)) \end{array}$$

Now let Φ be an arbitrary modal logic formula.
$\Pi_{\mathrm{F}}(\Phi, u)$ is inductively defined by[1]:

$$\begin{array}{lcl}
\Pi_{\mathrm{F}}(P(t_1,\ldots,t_n),u) & = & P'(u,\Pi_{\mathrm{F}}(t_1,u),\ldots,\Pi_{\mathrm{F}}(t_n,u)) \\
\Pi_{\mathrm{F}}(\neg\Phi,u) & = & \neg\Pi_{\mathrm{F}}(\Phi,u) \\
\Pi_{\mathrm{F}}(\Phi\vee\Psi,u) & = & \Pi_{\mathrm{F}}(\Phi,u)\vee\Pi_{\mathrm{F}}(\Psi,u) \\
\Pi_{\mathrm{F}}(\Phi\wedge\Psi,u) & = & \Pi_{\mathrm{F}}(\Phi,u)\wedge\Pi_{\mathrm{F}}(\Psi,u) \\
\Pi_{\mathrm{F}}(\forall x{:}\,D\ \Phi,u) & = & \forall x{:}\,D\ \Pi_{\mathrm{F}}(\Phi,u) \\
\Pi_{\mathrm{F}}(\exists x{:}\,D\ \Phi,u) & = & \exists x{:}\,D\ \Pi_{\mathrm{F}}(\Phi,u) \\
\Pi_{\mathrm{F}}(\Box\Phi,u) & = & \forall v{:}\,W\ R(u,v)\Rightarrow\Pi_{\mathrm{F}}(\Phi,v) \\
\Pi_{\mathrm{F}}(\Diamond\Phi,u) & = & \exists v{:}\,W\ R(u,v)\wedge\Pi_{\mathrm{F}}(\Phi,v)
\end{array}$$

The initial call for the translation of an arbitrary formula Φ is then $\Pi_{\mathrm{F}}(\Phi,\iota)$, where ι denotes the initial (or actual) world.

Definition 2.3 (The Interpretation Morphism $\Pi_{\Im}$)
Let $\Im_{\mathrm{M}} = (\mathcal{D}, \mathcal{F}_{\mathrm{M}}, \Im_{\mathrm{loc}}, \tau, \phi)$ be an interpretation over the signature Σ_{M}.
For any function symbol f in Σ_{M} let $\Im_{\mathrm{loc}}(\tau)(f) = \hat{f}_\tau$ and define $\hat{f}$ as:

$$\hat{f}(\tau, t_1, \ldots, t_n) = \hat{f}_\tau(t_1, \ldots, t_n)$$

Analogously for any predicate symbol P let $\Im_{\mathrm{loc}}(\tau)(P) = \hat{P}_\tau$ and define

$$\hat{P}(\tau, t_1, \ldots, t_n) = \hat{P}_\tau(t_1, \ldots, t_n)$$

Now let $\mathcal{M}$ be a $\Pi_\Sigma(\Sigma_{\mathrm{M}})$-structure with:

- $\mathcal{M}(f') = \hat{f}$
- $\mathcal{M}(P') = \hat{P}$
- $\mathcal{M}(R) = \Re$
- $\mathcal{M}(\iota) = \tau$
- $\mathcal{M}(D) = \mathcal{D}$
- $\mathcal{M}(W) = \mathcal{W}$

Then we define: $\Pi_{\Im}(\Im_{\mathrm{M}}) = (\mathcal{M}, \phi)$.
We usually call $\Pi_{\Im}(\Im_{\mathrm{M}})$ *the (classical) interpretation generated from* $\Im_{\mathrm{M}}$.

As we now have a translation from first-order modal logic into first-order predicate logic, we have to show that the translation indeed behaves as desired, i.e. we have to show that whenever a modal logic formula has a model then also its translation has one and vice versa. The corresponding proofs are performed by induction over the structure of modal logic formulae. To this end the following auxiliary lemma turns out to be useful:

[1]Note that we have to consider a two-sorted predicate logic because of the two sorts W and D.

Lemma 2.4
Let $\Im_{\mathrm{M}} = (\mathcal{D}, (\mathcal{W}, \Re), \Im_{\mathrm{loc}}, \tau, \phi)$ be an interpretation over the signature Σ_{M}, Φ be a modal logic formula, χ be a world from $\mathcal{W}$, and u be some world term. Then

$$\Pi_{\Im}(\Im_{\mathrm{M}}[\chi]) \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi, \iota) \text{ iff } \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi] \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi, u)$$

Proof: First we have to show that the respective interpretations of terms are identical.
This will be done as usual by induction over the term structure:

$$\begin{array}{rcl}
\Pi_{\Im}(\Im_{\mathrm{M}}[\chi])(\Pi_{\mathrm{F}}(x, \iota)) & = & \phi(x) \\
 & = & \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi](\Pi_{\mathrm{F}}(x, u)) \\
\Pi_{\Im}(\Im_{\mathrm{M}}[\chi])(\Pi_{\mathrm{F}}(t, \iota)) & = & \Pi_{\Im}(\Im_{\mathrm{M}}[\chi])(\Pi_{\mathrm{F}}(f(\ldots, t_i, \ldots), \iota)) \\
 & = & \Pi_{\Im}(\Im_{\mathrm{M}}[\chi])(f'(\iota, \ldots, \Pi_{\mathrm{F}}(t_i, \iota), \ldots)) \\
 & = & \hat{f}(\chi, \ldots, \Pi_{\Im}(\Im_{\mathrm{M}}[\chi])(\Pi_{\mathrm{F}}(t_i, \iota)) \ldots) \\
 & = & \hat{f}(\chi, \ldots, \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi](\Pi_{\mathrm{F}}(t_i, u)), \ldots) \\
 & & \text{by induction hypothesis} \\
 & = & \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi](f'(u, \ldots, \Pi_{\mathrm{F}}(t_i, u), \ldots)) \\
 & = & \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi](\Pi_{\mathrm{F}}(f(\ldots, t_i, \ldots), u))
\end{array}$$

Now we can start to prove the lemma for arbitrary modal logic formulae.
Note that if Φ is an atom then the proof works similar to the case of the complex terms from above. Also if $\Phi = \neg\Psi$ or $\Phi = \Psi_1 \circ \Psi_2$, where $\circ$ is any classical logical connective, there is no problem at all. Even if $\Phi = \forall x{:}\, D\ \Psi$ or if $\Phi = \exists x{:}\, D\ \Psi$ no difficulties do occur.
Therefore consider the case where $\Phi = \Box\Psi$:

$$\begin{array}{ll}
\multicolumn{2}{l}{\Pi_{\Im}(\Im_{\mathrm{M}}[\chi]) \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Box\Psi, \iota)} \\
\text{iff} & \Pi_{\Im}(\Im_{\mathrm{M}}[\chi]) \models_{\mathrm{PL}} \forall v{:}\, W\ (R(\iota, v) \Rightarrow \Pi_{\mathrm{F}}(\Psi, v)) \\
\text{iff} & \Pi_{\Im}(\Im_{\mathrm{M}}[\chi])[v/\xi] \models_{\mathrm{PL}} R(\iota, v) \Rightarrow \Pi_{\mathrm{F}}(\Psi, v) \\
 & \text{for any world } \xi \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Pi_{\Im}(\Im_{\mathrm{M}}[\chi])[v/\xi] \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Psi, v) \\
 & \text{for any world } \xi \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Pi_{\Im}(\Im_{\mathrm{M}}[\chi][\xi]) \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Psi, \iota) \\
 & \text{by induction hypothesis} \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Pi_{\Im}(\Im_{\mathrm{M}}[\xi]) \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Psi, \iota) \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Pi_{\Im}(\Im_{\mathrm{M}})[v/\xi] \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Psi, v) \\
 & \text{by induction hypothesis}
\end{array}$$

$$
\begin{array}{ll}
\text{iff} & \Re(\chi,\xi) \text{ implies } \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi][v/\xi] \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Psi,v) \\
& u \text{ is not free in } \Pi_{\mathrm{F}}(\Psi,v) \\
\text{iff} & \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi][v/\xi] \models_{\mathrm{PL}} R(u,v) \Rightarrow \Pi_{\mathrm{F}}(\Psi,v) \\
\text{iff} & \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi] \models_{\mathrm{PL}} \forall v{:}\, W\ (R(u,v) \Rightarrow \Pi_{\mathrm{F}}(\Psi,v)) \\
\text{iff} & \Pi_{\Im}(\Im_{\mathrm{M}})[u/\chi] \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Box\Psi,u)
\end{array}
$$

and similarly for $\Phi = \Diamond\Psi$

Thus soundness of the translation can be proved as follows:

Lemma 2.5
Let $\Im_{\mathrm{M}}$ be an interpretation over Σ_{M}, and Ψ a modal logic formula. Then

$$\Im_{\mathrm{M}} \models_{\mathrm{M}} \Psi \text{ iff } \Pi_{\Im}(\Im_{\mathrm{M}}) \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Psi,\iota)$$

Proof: As usual we first have to consider the evaluation of terms. Therefore we have to show that $\Im_{\mathrm{M}}(t) = \Pi_{\Im}(\Im_{\mathrm{M}})(\Pi_{\mathrm{F}}(t,\iota))$ for an arbitrary term t.

$$
\begin{array}{lcl}
\Im_{\mathrm{M}}(x) & = & \phi(x) \\
& = & \Pi_{\Im}(\Im_{\mathrm{M}})(\Pi_{\mathrm{F}}(x,\iota)) \\
\Im_{\mathrm{M}}(f(\ldots,t_i,\ldots)) & = & \hat{f}(\chi,\ldots,\Pi_{\Im}(\Im_{\mathrm{M}})(\Pi_{\mathrm{F}}(t_i,\iota))\ldots) \\
& & \text{by induction hypothesis} \\
& = & \Pi_{\Im}(\Im_{\mathrm{M}})(f'(\iota,\ldots,\Pi_{\mathrm{F}}(t_i,\iota),\ldots)) \\
& = & \Pi_{\Im}(\Im_{\mathrm{M}})(\Pi_{\mathrm{F}}(f(\ldots,t_i,\ldots),\iota))
\end{array}
$$

Thus the relational translation behaves as desired on the evaluation of terms.
With that we can now prove the lemma by induction on the structure of Ψ.
Note that the base case (where Ψ is a literal) and the cases where Ψ is composed by two fomulae and a classical logical connective are again obvious. Also there are no problems if Ψ is a quantified formula.
Therefore consider the case where $\Psi = \Box\Phi$:

$$
\begin{array}{ll}
\multicolumn{2}{l}{\Im_{\mathrm{M}} \models_{M} \Box\Phi} \\
\text{iff} & \Re(\tau,\chi) \text{ implies } \Im_{\mathrm{M}}[\chi] \models_{\mathrm{M}} \Phi \\
& \text{for any world } \chi \\
\text{iff} & \Re(\tau,\chi) \text{ implies } \Pi_{\Im}(\Im_{\mathrm{M}}[\chi]) \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi,\iota) \\
& \text{by induction hypothesis} \\
\text{iff} & \Re(\tau,\chi) \text{ implies } \Pi_{\Im}(\Im_{\mathrm{M}})[w/\chi] \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi,w) \\
& \text{by Lemma 2.4} \\
\text{iff} & \Pi_{\Im}(\Im_{\mathrm{M}})[w/\chi] \models_{\mathrm{PL}} R(\iota,w) \Rightarrow \Pi_{\mathrm{F}}(\Phi,w) \\
\text{iff} & \Pi_{\Im}(\Im_{\mathrm{M}}) \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Box\Phi,\iota)
\end{array}
$$

and analogous for $\Psi = \Diamond\Phi$

Hence relational translation is sound in the sense that for any interpretation satisfying a modal logic formula Φ there is a predicate logic interpretation satisfying the translated version of Φ. Or in other words, the translation preserves satisfiability. In particular, if the accessibility relation $\Re$ for the given interpretation obeys certain properties the corresponding generated predicate logic interpretation satisfies the respective translations.
Now it is necessary to show that also unsatisfiability is preserved, i.e. the translation is complete. To this end a "reverse-translation" is defined which generates a modal logic interpretation from a given predicate logic interpretation. Note that in proving the soundness of the reverse-translation the completeness of the actual translation is shown.

Definition 2.6 (The Mapping $\Pi_{\Im}^{-1}$)
Let $\Im_{\mathrm{PL}} = (\mathcal{M}, \phi)$ be a classical interpretation over the signature $\Pi_\Sigma(\Sigma_\mathrm{M})$.
Let furthermore $\Im_\mathrm{M} = (\mathcal{D}, (\mathcal{W}, \Re), \Im_{\mathrm{loc}}, \tau, \phi)$ be an interpretation over Σ_M where:

- $\Re = \mathcal{M}(R)$
- $\mathcal{W} = \mathcal{M}(W)$
- $\mathcal{D} = \mathcal{M}(D)$
- $\tau = \mathcal{M}(\iota)$
- for any τ' in $\mathcal{W}$: $\Im_{\mathrm{loc}}(\tau')(f) = \hat{f}_{\tau'}$ and $\Im_{\mathrm{loc}}(\tau')(P) = \hat{P}_{\tau'}$

Then we call $\Im_\mathrm{M} = \Pi_{\Im}^{-1}(\Im_{\mathrm{PL}})$ the (modal logic) interpretation *generated from* $\Im_{\mathrm{PL}}$.

Lemma 2.7
Let $\Im_{\mathrm{PL}}$ be a classical interpretation over the signature $\Pi_\Sigma(\Sigma_\mathrm{M})$, and $\chi \in \mathcal{M}(W)$. Then for any modal logic formula Φ:

$$\Im_{\mathrm{PL}}[u/\chi] \models_{\mathrm{PL}} \Pi_F(\Phi, u) \text{ iff } \Pi_{\Im}^{-1}(\Im_{\mathrm{PL}})[\chi] \models_\mathrm{M} \Phi$$

Proof: Works similar to the soundness proof in Lemma 2.5. It is thus sufficient to consider the critical case where $\Phi = \Box\Psi$.

$$\begin{array}{ll}
\multicolumn{2}{l}{\Im_{\mathrm{PL}}[u/\chi] \models_{\mathrm{PL}} \Pi_\mathrm{F}(\Phi, u)} \\
\text{iff} & \Im_{\mathrm{PL}}[u/\chi] \models_{\mathrm{PL}} \forall v{:}\, W\ R(u, v) \Rightarrow \Pi_\mathrm{F}(\Psi, v) \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Im_{\mathrm{PL}}[u/\chi][v/\xi] \models_{\mathrm{PL}} \Pi_\mathrm{F}(\Psi, v) \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Im_{\mathrm{PL}}[v/\xi] \models_{\mathrm{PL}} \Pi_\mathrm{F}(\Psi, v) \\
 & \text{since } u \text{ is not free in } \Pi_\mathrm{F}(\Psi, v) \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Pi_{\Im}^{-1}(\Im_{\mathrm{PL}})[\xi] \models_\mathrm{M} \Psi \\
 & \text{by the induction hypothesis} \\
\text{iff} & \Re(\chi, \xi) \text{ implies } \Pi_{\Im}^{-1}(\Im_{\mathrm{PL}})[\chi][\xi] \models_\mathrm{M} \Psi \\
\text{iff} & \Pi_{\Im}^{-1}(\Im_{\mathrm{PL}})[\chi] \models_\mathrm{M} \Box\Psi
\end{array}$$

Corollary 2.8
Let $\Im_{\mathrm{PL}}$ be a classical model for $\Pi_\mathrm{F}(\Phi, \iota)$. Then $\Pi_{\Im}^{-1}(\Im_{\mathrm{PL}})$ is a model for Φ.
Proof: Follows easily from Lemma 2.7 if u is set to ι and χ is set to $\mathcal{M}(\iota) = \tau$.

Thus both soundness and completeness of relational translation have been shown. Now assume that there is a modal logic formula Φ to be proved valid in some particular modal logic, say KT4. This can be performed by proving the unsatisfiability of $\neg\Phi$. In order to do this the relational translation technique allows to translate $\neg\Phi$ into $\Psi = \Pi_F(\neg\Phi, \iota)$ and to prove the unsatisfiability of Ψ instead, provided the necessary additional axioms of the modal logic under consideration are added. In this example the respective axioms T and 4 have to be included, i.e. $\forall x\ R(x,x)$ and $\forall x, y, z\ R(x,y) \wedge R(y,z) \Rightarrow R(x,z)$. The following theorem fixes this in general. It is hereby assumed that ML is an arbitrary modal logic as e.g. KDB4, and that ML* represents the set of corresponding additional axioms (which are seriality, symmetry, and transitivity in the case of KDB4).

Theorem 2.9
Φ is a theorem of the modal logic ML if and only if any predicate logic interpretation satisfying the ML-Axioms is a model for $\Pi_{\mathrm{F}}(\Phi, \iota)$. Or, more formally:

$$\models_{\mathrm{ML}} \Phi \text{ iff } \mathrm{ML}^* \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi, \iota)$$

Proof: Assume $\models_{\mathrm{ML}} \Phi$ and assume further that there exists a predicate logic interpretation which satisfies the axioms of ML* but not the formula $\Pi_{\mathrm{F}}(\Phi, \iota)$. Then this PL-interpretation satisfies $\Pi_{\mathrm{F}}(\neg\Phi, \iota)$. However, Corollary 2.8 then guarantees the existence of a modal logic interpretation which satisfies $\neg\Phi$ and this contradicts the assumption that $\models_{\mathrm{ML}} \Phi$.
On the other hand assume that every predicate logic interpretation which is a model for the ML*-Axioms also satisfies $\Pi_{\mathrm{F}}(\Phi, \iota)$ and that there is a modal logic interpretation which is an ML-model for $\neg\Phi$. But then there exists a predicate logic interpretation satisfying the ML*-Axioms which is model for $\Pi_{\mathrm{F}}(\neg\Phi, \iota)$ which again contradicts the assumption.

Chapter 3

Functional Simulation

The main idea behind functional simulation is the replacement of binary predicates inside formulae by special equations. Under certain circumstances it is then possible to eliminate even these equations. Such circumstances do in fact have a syntactic counterpart, i.e. a characteristic kind of occurrence.
However, introduction of such equations requires additional sorts to be introduced. These sorts are called *functional simulators* and consist of (as the name suggests) functions.

Definition 3.1 (Functional Simulators)
Let S and T be two non-empty denumerable sets and let R be a non-empty binary relation over $S \times T$.
Then define for any pair (x, y) in $S \times T$: $(x, y) \approx (u, v)$ iff $x = u$.
Obviously $\approx$ denotes an equivalence relation. It is thus possible to introduce equivalence classes $[\,]_{\approx}$ by:

$$[(x, y)]/_{\approx} = \{(u, v) \in R \mid (x, y) \approx (u, v)\}$$

and

$$R/_{\approx} = \{[(x, y)]/_{\approx} \mid (x, y) \in R\}$$

Both $[(x, y)]/_{\approx}$ and $R/_{\approx}$ are denumerable, therefore there exist surjective mappings $\theta : \mathrm{Nat} \to R/_{\approx}$ and $\delta_i : \mathrm{Nat} \to \theta(i)$.
Then define $f_j = \{\delta_k(j) \mid k \in \mathrm{Nat}\}$. We call $F_R = \{f_j \mid j \in \mathrm{Nat}\}$ a *functional simulator* of R on $S \times T$.

The following definition turns out to be useful for the sequel:

Definition 3.2
Let $R \subseteq S \times T$. An element $s \in S$ is called *normal w.r.t.* R if there exists a $t \in T$ such that $(s, t) \in R$.

Some of the properties of functional simulators are given by the following lemma:

Lemma 3.3
Let R be a binary relation over $S \times T$, where both S and T are denumerable and let F_R be a functional simulator of R on $S \times T$. Then

- F_R denotes a denumerable set of partial functions from S to T
- If x is normal w.r.t. R then any $f \in F_R$ is defined on x.
- Thus, if R is left-total, i.e. any $x \in S$ is normal w.r.t. R,
- If R is left-total then for any $u \in S$ and any $f \in F_R$: $R(u, f(u))$ then F_R denotes a denumerable set of total functions from S to T.

and for any $u \in S$ and any $v \in T$

- If $R(u, v)$ then there exists an $f \in F_R$ such that $f(u) = v$
- If u is normal w.r.t. R then for any $f \in F_R$: $R(u, f(u))$

Proof: Obvious by construction.

Recall that by definition 1.7 we have that:

$$\Im \models_{\mathrm{M}} \Diamond\Phi \quad \text{iff} \quad \Re(\tau, \chi) \text{ and } \Im[\chi] \models_{\mathrm{M}} \Phi \text{ for some } \chi \in \mathcal{W}$$

Since $\Re \subseteq \mathcal{W} \times \mathcal{W}$ there exists a functional simulator $F_\Re$ for $\Re$ on $\mathcal{W} \times \mathcal{W}$ with properties given in Lemma 3.3. The above thus implies:

$$\tau \text{ is normal and for some } f \in F_\Re \; f(\tau) = \chi \text{ and } \Im[\chi] \models_{\mathrm{M}} \Phi$$

This on the other hand can be simplified to:

$$\tau \text{ is normal and for some } f \in F_\Re\text{: } \Im[f(\tau)] \models_{\mathrm{M}} \Phi$$

since χ does not occur in Φ.
Recall again the two steps from above: firstly, the positive occurrence of the accessibility relation $\Re$ has been replaced by some equation and secondly, this equation could be eliminated by some easy simplification step.
It is thus possible to define a new formula morphism, namely one which performs exactly the above. This certainly has its effect also on the signature morphism. We therefore assume two more non-logical symbols: F_R and N which denote the sort given by the functional simulator of $\Re$ and the predicate "Normal" respectively.

Definition 3.4 (Negation Normal Form)
A formula Φ is in *negation normal form* if it contains no implication or equivalence and all negations occur solely in front of the atoms.

Obviously any arbitrary formula can be transformed into an equivalent one which is in negation normal form. We will assume in the sequel that any formula is in negation normal form. This is reasonable since the change in the definition of the formula morphism Π_{F} will treat $\Box$-formulae different to $\Diamond$-formulae. The duality of $\Box$ and $\Diamond$ would therefore cause problems without something like a negation normal form.

Definition 3.5 (The Formula Morphism $\Pi^\star_{\mathrm{F}}$)
Let Φ be a modal logic formula in negation normal form.

$$\Pi^\star_{\mathrm{F}}(\Diamond\Phi, u) = N(u) \wedge \exists f\colon F_R\, \Pi^\star_{\mathrm{F}}(\Phi, f(u))$$

In any other case $\Pi^\star_{\mathrm{F}}$ behaves as Π_{F}.

Remark 3.6 Although the quantification over functions suggests a second-order quantification we indeed have not got beyond classical (many-sorted) first-order predicate logic. The reason is that we in fact do not consider all functions which map worlds to worlds but, as Lemma 3.3 shows, merely a denumerable subset of those. Hence it is possible to view a functional simulator just as a usual sort.

Although the results of $\Pi^\star_{\mathrm{F}}$ are usually of smaller size than the results of Π_{F} they become rather unreadable. We therefore use a slightly different notation, the so-called *world-paths*. I.e. we assume a function “apply” which accepts two arguments, namely a function, say f, and an element of the function's domain, say x, and results in the value of $f(x)$. Any term of the form $f_n(f_{n-1}(\ldots f_1(x)\ldots))$ thus becomes $\mathrm{apply}(f_n, \mathrm{apply}(f_{n-1}, \ldots, \mathrm{apply}(f_1, x)\ldots))$ which again is simplified to the world-path $[x\, f_1\, \ldots, f_n]$. As an example consider the translation of the modal logic formula $\Diamond\Diamond P$.

$$\begin{array}{rcl}\Pi^\star_{\mathrm{F}}(\Diamond\Diamond P, \iota) & = & \exists x \exists y\; N(\iota) \wedge N(x(\iota)) \wedge P'(y(x(\iota))) \\ & = & \exists x \exists y\; N([]) \wedge N([x]) \wedge P'([x\, y])\end{array}$$

Note that ι itself is thus represented by the “empty world-path” [ ] then. This notation has two advantages; firstly, it is a bit more compact and secondly, the variables appearing in world-paths directly correspond to the modal operators of the translated formula and that even in the order they occur there. Sometimes we even omit the square brackets if it is obvious where they are supposed to be.

Definition 3.7 (Simulator Axiom)
According to Lemma 3.3 we call the formula $\mathrm{Sim} = \mathrm{Sim}_1 \wedge \mathrm{Sim}_2$ the *simulator axiom* for R where

$$\begin{array}{rcl}\mathrm{Sim}_1 & = & \forall u, v\colon W\; R(u, v) \Rightarrow \exists x\colon F_R\; [u\, x] = v \\ \mathrm{Sim}_2 & = & \forall u\colon W\; \forall x\colon F_R\; N(u) \Rightarrow R(u, ux)\end{array}$$

Theorem 3.8
Let ML be a modal logic and let Φ be a modal logic formula. Then

$$\models_{\mathrm{ML}} \Phi \quad \text{iff} \quad \mathrm{ML}^* \wedge \mathrm{Sim}_2 \models_{\mathrm{PL}} \Pi^\star_{\mathrm{F}}(\Phi, \iota)$$

where ML^* denotes the correspondence axioms for the modal logic ML.
Proof: By Theorem 2.9 we know that $\models_{\mathrm{ML}} \Phi$ iff $\mathrm{ML}^* \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi, \iota)$. This, on the other hand, is equivalent to $\models_{\mathrm{ML}} \Phi$ iff $\mathrm{ML}^* \wedge \mathrm{Sim} \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi, \iota)$ because for every binary relation there exists a functional simulator[1].
Furthermore it can easily be shown that every predicate logic interpretation over the signature $\Pi_{\Sigma}(\Sigma_{\mathrm{M}})$ which is a model for Sim also satisfies the formula $\Pi_{\mathrm{F}}(\Phi, u) \Leftrightarrow \Pi^{\star}_{\mathrm{F}}(\Phi, u)$. Therefore $\mathrm{ML}^* \wedge \mathrm{Sim} \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi, u) \Leftrightarrow \Pi^{\star}_{\mathrm{F}}(\Phi, u)$. Hence $\mathrm{ML}^* \wedge \mathrm{Sim} \models_{\mathrm{PL}} \Pi_{\mathrm{F}}(\Phi, \iota)$ iff $\mathrm{ML}^* \wedge \mathrm{Sim} \models_{\mathrm{PL}} \Pi^{\star}_{\mathrm{F}}(\Phi, \iota)$.
Since the set of interpretations satisfying $\mathrm{ML}^* \wedge \mathrm{Sim}$ is a subset of the set of interpretations satisfying $\mathrm{ML}^* \wedge \mathrm{Sim}_2$ the only thing which remains to be shown is:

$$\mathrm{ML}^* \wedge \mathrm{Sim} \models_{\mathrm{PL}} \Pi^{\star}_{\mathrm{F}}(\Phi, \iota) \text{ implies } \mathrm{ML}^* \wedge \mathrm{Sim}_2 \models_{\mathrm{PL}} \Pi^{\star}_{\mathrm{F}}(\Phi, \iota)$$

To this end we show that if $\mathrm{ML}^* \wedge \mathrm{Sim}_2 \wedge \Pi^{\star}_{\mathrm{F}}(\Psi, \iota)$ is satisfiable then also $\mathrm{ML}^* \wedge \mathrm{Sim} \wedge \Pi^{\star}_{\mathrm{F}}(\Psi, \iota)$ is. Therefore assume that $\mathrm{ML}^* \wedge \mathrm{Sim}_2 \wedge \Pi^{\star}_{\mathrm{F}}(\Psi, \iota)$ is satisfiable but $\Xi = \mathrm{ML}^* \wedge \mathrm{Sim} \wedge \Pi^{\star}_{\mathrm{F}}(\Psi, \iota)$ is not. Then there exists a ground instance of the clause normal form for Ξ such that each path through it is unsatisfiable. In particular, all those paths leading through the ground instances of the literal $uf = v$ which stems from Sim_1 have to be unsatisfiable. However, nowhere in the clause set there is another occurrence of the symbol f, thus this literal and therefore the whole clause is superfluous and can be omited. From this it follows what has been claimed.
Now, if every interpretation which satisfies $\mathrm{ML}^* \wedge \mathrm{Sim}$ is a model for $\Pi^{\star}_{\mathrm{F}}(\Phi, \iota)$ then also each interpretation which satisfies $\mathrm{ML}^* \wedge \mathrm{Sim}_2$ because otherwise for some interpretation $\Im$ it holds that $\Im \models_{\mathrm{PL}} \mathrm{ML}^* \wedge \mathrm{Sim}_2$ but $\Im$ is no model for $\Pi^{\star}_{\mathrm{F}}(\Phi, \iota)$. By the above there is an interpretation $\Im'$ then which satisfies $\mathrm{ML}^* \wedge \mathrm{Sim}_2$ and which is not a model for $\Pi^{\star}_{\mathrm{F}}(\Phi, \iota)$. But this contradicts our assumption and we are done.

Summarizing: assume there is a formula Φ to be proved valid for modal logic ML. If the modal logic ML is not just K then there do exist some correspondence axioms ML^* which reflect the properties of the underlying accessibility relation. As it has been shown in Theorem 3.8 proving the ML-validity of Φ is tantamount to proving the validity of the first-order predicate logic formula $\mathrm{ML}^* \wedge \mathrm{Sim}_2 \wedge \Pi^{\star}_{\mathrm{F}}(\Phi, \iota)$.

[1]Of course, this actually requires a further logic morphism from the given predicate logic to another predicate logic with an additional sort for the functional simulator. Nevertheless, this can be done quite easily and is therefore omitted here.

Chapter 4

Horn Clauses

The following definition is the usual one for Horn Clauses:

Definition 4.1 (Horn Clauses)
A first-order predicate logic formula is called *in Horn form* iff each clause of its clause normal form contains at most one positive literal.

Note that each of the correspondence axioms for the modal logics we consider is Horn.

At this stage it looks as if it was already possible to utilize PROLOG as a theorem proving tool for modal logics. However, a simple examination shows that the particular way PROLOG is dealing with formulae often leads to infinite loops under the occurrence of certain theory axioms as e.g. symmetry or transitivity. Such loops can sometimes be omitted by wisely ordering the theory axioms, although this is not always possible. At least in cases where a non-theorem is tried to be proved, PROLOG runs into loops, regardless of whether the modal logic is decidable or not.

Example 4.2
Consider the formula $\Box P \Rightarrow \Diamond P$ which is not valid for the modal logic KB. Running PROLOG as a theorem prover means to negate the formula, derive the clause normal form and add the additional clause for Symmetry. The resulting PROLOG program then is:

$$\begin{array}{lcl} P(u) & \leftarrow & R(\iota, u) \\ R(u, ux) & \leftarrow & N(u) \\ R(u, v) & \leftarrow & R(v, u) \\ & \leftarrow & P(v), R(\iota, v) \end{array}$$

It can immediately be seen that PROLOG runs into an endless loop although the modal logic KB is in fact decidable.

In the following chapter some simplifications will be presented which not only avoid these kinds of loops but also simplify the inference mechanism considerably.

A further remark should be added: The translation approach introduced results in formula of many-sorted first-order predicate logic although PROLOG cannot distinguish between different sorts, unless they are distinguished explicitly by some extra sort predicate symbols. So, how can standard PROLOG be utilized? The simple answer is: we neither need to include unary sort predicate symbols nor do we need to extend standard PROLOG in order to be able to deal with these sorts because of the particular structure of the translation results. Variables of different sorts can only occur on certain positions inside clauses and these positions cannot be mixed up by unification. Therefore PROLOG does not have to be able to distinguish explicitly between diffferent sorts; in some sense it does so implicitly.

Chapter 5

Simplifications

In this chapter some of the most useful simplifications will be introduced. They are based partly on syntactic, partly on semantic grounds.

5.1 Serial Modal Logics

Applying the functional simulation approach to binary predicate symbols like the accessibility relation symbol for modal logics has the advantage that the frequent occurrence of pure R-clauses is totally avoided. This in fact reduces the size of the search space considerably.
For serial modal logics it is even the case that the number of clauses is reduced even more. This is stated by the following lemma:

Definition 5.1
Let Φ be a modal logic formula. By flat(Φ) we understand the first-order predicate logic formula we gain from Φ after ignoring the modal operators, i.e. we define:

$$\begin{array}{rcl} \text{flat}(\Box\Phi) & = & \text{flat}(\Phi) \\ \text{flat}(\Diamond\Phi) & = & \text{flat}(\Phi) \end{array}$$

For all the other cases "flat" is just the usual homomorphic extension.

Lemma 5.2
Let Φ be a (serial) modal logic formula in negation normal form.

1. $\Pi^\star_{\text{F}}(\Phi, u)$ is Horn iff flat(Φ) is Horn (for arbitrary u)
2. The clause normal form of $\Pi^\star_{\text{F}}(\Phi, \iota)$ contains exactly as many clauses as the clause normal form of flat(Φ)

Proof: The first part is proved by induction over the structure of Φ:
Base case: Φ is a literal. This is obviously trivial
Induction step: There is no problem if $\Phi = \forall x\, \Psi$ or $\Phi = \exists x\, \Psi$. Therefore proofs for the following cases remain to be performed:

1. $\Phi = \Phi_1 \wedge \Phi_2$
flat(Φ) is Horn if and only if flat(Φ_1) is Horn and flat(Φ_2) is Horn. This is equivalent to $\Pi^\star_{\mathrm{F}}(\Phi_1, u)$ is Horn and $\Pi^\star_{\mathrm{F}}(\Phi_2, u)$ is Horn by induction hypothesis and thus equivalent to $\Pi^\star_{\mathrm{F}}(\Phi, u)$ is Horn.

2. $\Phi = \Phi_1 \vee \Phi_2$
Call a formula *clause-negative* if its clause normal form contains no positive literal.
flat(Φ) is Horn if and only if flat(Φ_1) is Horn and flat(Φ_2) is Horn and either flat(Φ_1) is clause-negative or flat(Φ_2) is clause-negative (or both). Suppose w.l.o.g. that flat(Φ_1) is clause-negative. An easy induction shows that $\Pi^\star_{\mathrm{F}}(\Phi_1, u)$ is then clause-negative as well. The desired result thus follows by induction hypothesis.

3. $\Phi = \Box\Psi$

$$\begin{array}{lll} \text{flat}(\Phi) \text{ is Horn} & \text{iff} & \text{flat}(\Psi) \text{ is Horn} \\ & \text{iff} & \Pi^\star_{\mathrm{F}}(\Psi, v) \text{ is Horn for any } v \\ & \text{iff} & \forall v\ R(u,v) \Rightarrow \Pi^\star_{\mathrm{F}}(\Psi, v) \text{ is Horn} \\ & \text{iff} & \Pi^\star_{\mathrm{F}}(\Phi, u) \text{ is Horn} \end{array}$$

4. $\Phi = \Diamond\Psi$

$$\begin{array}{lll} \Pi^\star_{\mathrm{F}}(\Diamond\Psi, u) \text{ is Horn} & \text{iff} & \Pi^\star_{\mathrm{F}}(\Psi, v) \text{ is Horn} \\ & \text{iff} & \text{flat}(\Psi) \text{ is Horn} \\ & \text{iff} & \text{flat}(\Phi) \text{ is Horn} \end{array}$$

For the second part of the lemma note that *seriality* means that every world is normal. Thus $\forall u{:}W\ \ N(u)$ holds. With this the translation morphism $\Pi^\star_{\mathrm{F}}$ can slightly be changed to:

$$\Pi^\star_{\mathrm{F}}(\Diamond\Phi, u) = \exists x{:}\, F_R\, \Pi^\star_{\mathrm{F}}(\Phi, ux)$$

and the correctness of the second claim of the lemma is evident.

Thus our translation approach results in clause sets which look very similar to the clauses we would get if we ignored the modal operators at all. The only source of inefficiency remaining is the set of theory axioms which stem from the particular modal logic under consideration. However, it turns out that in many cases these theory axioms can be significantly simplified. Some of these simplifications make use of particular structure modal formulae get after translation, namely that they do not contain any positive R-literal. We fix this in the following Lemma:

Lemma 5.3
Let Φ be an arbitrary modal logic formula in negation normal form and let t be an arbitrary world-path. Then the clause normal form of $\Pi^\star_{\mathrm{F}}(\Phi, t)$ contains no positive R-literal.

Proof: Easy induction on the structure of Φ. The only critical cases are those where Φ has a modal operator as a top symbol. So assume $\Phi = \Box\Psi$.
$\Pi^\star_{\mathrm{F}}(\Phi, t) = \forall u\ R(t, u) \Rightarrow \Pi^\star_{\mathrm{F}}(\Psi, u)$ by definition of the formula morphism $\Pi^\star_{\mathrm{F}}$. Thus the clause normal form of $\Pi^\star_{\mathrm{F}}(\Phi, t)$ contains no positive R-literals if and only if $\Pi^\star_{\mathrm{F}}(\Psi, u)$ contains no positive R-literals and this holds because of the induction hypothesis.
Now consider the other case where $\Phi = \Diamond\Psi$.
Then $\Pi^\star_{\mathrm{F}}(\Phi, t) = \exists x{:}\, F_R\ \Pi^\star_{\mathrm{F}}(\Psi, tx)$ which also contains no positive R-literals according to the induction hypothesis and we are done.

In the following sections this property will play an important rôle on simplification. In fact these are not merely simplifications but also some sort of transformations necessary in order to be able to utilize PROLOG as a modal logic inference machine, for the non-simplified versions often lead to infinite loops because of the particular PROLOG control. As it will be shown, the simplification avoids such possible loops.

5.1.1 The Modal Logic KD

KD is the “smallest” serial modal logic with Kripke-style possible world semantics. From the above we know that the formula morphism $\Pi^\star_{\mathrm{F}}$ can be altered to:

$$\begin{array}{lcl} \Pi^\star_{\mathrm{F}}(\Box\Phi, u) & = & \forall v{:}\, W\ R(u, v) \Rightarrow \Pi^\star_{\mathrm{F}}(\Phi, v) \\ \Pi^\star_{\mathrm{F}}(\Diamond\Phi, u) & = & \exists v{:}\, F_R\ \Pi^\star_{\mathrm{F}}(\Phi, [uv]) \end{array}$$

Since every world is normal (i.e. $\forall u{:}\, W\ N(u)$) also the simulator axiom Sim_2 can be simplified, namely to: $\forall u{:}\, W\, \forall x{:}\, F_R\ R(u, ux)$. Thus the theory clause for seriality, which is $\forall u{:}\, W\ R(u, [u\, f(u)])$, becomes superfluous. Its responsibilities are captured already by the simplified simulator axiom and no further theory axiom needs to be included.

Example 5.4
Consider the formula $\Box\Diamond\Box\Phi \Rightarrow \Diamond\Diamond\Box\Phi$ to be proved valid in the modal logic KD. The first thing to be done is to negate the theorem since PROLOG acts as a refutation procedure. The resulting formula then has to be translated by $\Pi^\star_{\mathrm{F}}$ into first-order predicate logic and we get:

$$\begin{array}{l} \Pi^\star_{\mathrm{F}}(\Box\Diamond\Box\Phi \wedge \Box\Box\Diamond\neg\Phi, \iota) \\ \qquad = \Pi^\star_{\mathrm{F}}(\Box\Diamond\Box\Phi, \iota) \wedge \Pi^\star_{\mathrm{F}}\Box\Box\Diamond\neg\Phi, \iota) \\ \qquad = \forall x\, \exists y\, \forall z\ \Phi([x\, y\, z]) \wedge \forall x\, \forall y\, \exists z\ \neg\Phi([x\, y\, z]) \end{array}$$

Transformation into clause normal form finally results in:

$$\Phi([x\, f(x)\, z]) \wedge \neg\Phi([u\, v\, g(u, uv)])$$

This is already in Horn form, thus the resulting PROLOG version is:

$$\begin{aligned}&\Phi([x\,f(x)\,z]).\\&\qquad\qquad\leftarrow \Phi([u\,v\,g(u,uv)]).\end{aligned}$$

Running this as a program is a trivial task for PROLOG, hence the formula can easily be proved valid in modal logic KD[1].

Those readers familiar with Hans Jürgen Ohlbach's thesis on resolution for modal logics might recognize this as the functional translation approach from modal logic to predicate logic. In fact, for the modal logic KD, there is actually no difference between the method proposed here and Ohlbach's approach (in the constant domain case). For other modal logics however, the two translation methods become different as can be seen e.g. in the next section.

5.1.2 The Modal Logic KDB

As shown already for the modal logic KD it is not necessary to include the correspondence axiom for seriality; its responsibilities are captured by the simplified simulator axiom $R(u, ux)$. Now, the modal logic KDB is based on serial and symmetric frames. Thus there is, in principle, the clause $R(u, v) \leftarrow R(v, u)$ (i.e. symmetry of R) to be added. We will now show how this additional clause can be simplified.

Suppose that during the PROLOG derivation the query $R(\alpha, \beta)$ occurs. There are two possibilities to solve this goal: either by the simulator axiom $R(u, ux)$ or by the symmetry clause. There is indeed no other possibility remaining because Lemma 5.3 guarantees that no other clause has an R-literal as a head. Now, the latter case produces a new goal, namely $R(\beta, \alpha)$. Again this new goal can only be solved either by $R(u, ux)$ or by the symmetry clause. But this time another application of the symmetry clause would not lead to anything new. Therefore this new goal indeed *has* to be solved by $R(u, ux)$. Since this case evaluation is complete we have found out that $R(\alpha, \beta)$ succeeds if and only if either $\alpha = [\beta\,x]$ or $\beta = [\alpha\,x]$ for some x (which is not necessarily a variable). Thus, instead of the symmetry clause $R(u, v) \leftarrow R(v, u)$ it is sufficient to take the unit clause $R(ux, u)$ in addition to $R(u, ux)$.
The following example shows the impact of this simplification.

Example 5.5
Consider the following formula to be proved valid for KDB:

$$\Diamond\Box\Diamond P \Rightarrow \Diamond P$$

[1]Note again that the world-path notation has been used merely for readability. In the actual PROLOG implementation the world-paths have to be expanded to the respective function applications.

Negation and transformation into Horn form results in the following Horn clause set:

$$\begin{array}{lcll}
P[u\,b(u)] & \leftarrow & R(a,u). & \text{From } \Diamond\Box\Diamond P \\
R(u,ux). & & & \text{The first theory clause} \\
R(ux,u). & & & \text{The second theory clause} \\
 & \leftarrow & P[v], R(\iota,v) & \text{From } \Diamond P
\end{array}$$

where a and b are skolem functions (constants).
Running PROLOG with this set of clauses results in the following derivation (brackets are omitted whereever their actual occurrence is obvious):

$$\begin{array}{ll}
\leftarrow & R(a,u), R(\iota,[u\,b(u)]) \\
\leftarrow & R(\iota,[a\,x\,b(ax)]) \quad \text{backtrack} \\
\leftarrow & R(\iota,b(\iota)) \\
\leftarrow & \text{yes}
\end{array}$$

In order to get a better understanding on why this translation fits much better to our desired purposes and is even necessary if we want to utilize PROLOG, we try to prove the above theorem with the help of the (unmodified) relational translation:

Example 5.6
Consider again the formula: $\Diamond\Box\Diamond P \Rightarrow \Diamond P$. Negation, relational translation and transformation into Horn form results in the following Horn clause set:

$$\begin{array}{lcl}
R(\iota,a). & & \\
R(x,b(x)) & \leftarrow & R(a,x). \\
P(b(x)) & \leftarrow & R(a,x). \\
R(u,v) & \leftarrow & R(v,u). \\
R(u,f(u)). & & \\
 & \leftarrow & P(y), R(\iota,y).
\end{array}$$

Running this as a PROLOG program results in an infinite loop. In fact, a closer examination of the clause set shows that there is no possible arrangement of the given clauses such that an infinite loop can be avoided, although the set of clauses is indeed unsatisfiable.

Note that the application of functional simulators guarantees that such cases of infinite loops are impossible.
Another remark: Even when relational translation without functional simulation does not run into a loop for given theorems, it will always loop for non-theorems. These endless loops are also avoided by the functional simulation approach.

5.1.3 The Modal Logic KD4

Similarily to the case of KDB we proceed with the modal logic KD4, i.e. the modal logic characterized by the serial and transitive frames. The correspondence axiom to be added is (in its PROLOG form):

$$R(u,w) \leftarrow R(u,v), R(v,w)$$

Now assume again a query $R(\alpha,\beta)$. It is easy to see that this query can be solved if and only if $\beta = [\alpha\, x_1, \ldots\, x_n]$ where the x_i are elements of F_R. Or, in other words, any ground $R(\alpha,\beta)$ can be solved iff α is a proper prefix of β.
The same property however, we get if we take the clause $R(u,vx) \leftarrow R(u,v)$ instead of the transitivity clause $R(u,w) \leftarrow R(u,v), R(v,w)$. Thus we have found a suitable simplification for the theory clauses, namely by adding the two clauses

$$\begin{array}{l} R(u,ux). \\ R(u,vx) \leftarrow R(u,v) \end{array}$$

to the clauses of the input formula. Note again that also this simplification is only possible because we can assume that there are no other clauses which contain positive R-Literals.

A closer examination of the simplications performed so far shows that the simplified clauses are in fact derivable from the original theory clauses by a single resolution step. This suggests that another resolution possibility also results in a suitable simplified clause for transitivity, namely $R(u,v) \leftarrow R(ux,v)$. But there is a big difference between this kind of simplification and the former one which has to do with the PROLOG control. Both behave well whenever the input formula is indeed a theorem of the logic KD4. On non-theorems however, their behaviour differs considerably because the clause $R(u,v) \leftarrow R(ux,v)$ will run into an endless loop whereas the first simplification will always terminate. Moreover, the second simplification indeed requires that the simulator axiom comes before the transitivity axiom, so there are quite some good reasons to prefer the first simplification $R(u,vx) \leftarrow R(u,v)$ over $R(u,v) \leftarrow R(ux,v)$.
Another thing should be remarked at this stage: Since the simplification we presented all were derivable from the theory axioms by a single resolution step we could equally try to simplify the already simplified version by another resolution step, which then results in: $R(u,uxy)$. This unit clause is obviously correct, but unfortunately it is not sufficient as a substitute for $R(u,vx) \leftarrow R(u,v)$ because not all possible self-resolutions steps have been taken into account. This was no problem with the first simplification because self-resolution didn't lead to anything new. Thus the only further "simplification" would be to add the (infinitly many) unit clauses $R(u,uxy)$ and $R(u,uxyz)$ and so on, but this obviously is impossible.

5.1.4 The Modal Logic S4 (or KT4)

For S4 it is (in analogy to KD4) possible to simplify the transitivity axiom represented by the clause $R(u,w) \leftarrow R(u,v), R(v,w)$ to $R(u,vx) \leftarrow R(u,v)$.
In addition the axiom $R(u,u)$ has to be added in order to guarantee reflexivity of the

underlying accessibility relation. Both together, however, imply $R(u, ux)$; thus this unit clause doesn't need to be included any more. The theory axioms thus become:

$$\begin{array}{lcl} R(u, u). & & \\ R(u, vx) & \leftarrow & R(u, v) \end{array}$$

5.1.5 The Modal Logic S5 (or KTB4, KDB4, KTE, KDBE)

This modal logic is determined by an accessibility relation which is an equivalence, i.e. any two worlds either access each other or there is no R-connection whatsoever between them[2]. However, it is a well known property of modal logics that the consideration of R-connected frames is sufficient. (This can be found, for instance, in [2] or in [9].) W.l.o.g. we can thus assume that, for S5, each world has access to any other world (and to itself of course). Hence we do not have to include all the correspondence axioms; the simple unit-clause $R(u, w)$ will do as well.
And again, since the whole theory on the accessibility relation R is given by a single unit clause, its effect can be directly put into the formula morphism, i.e. the mapping $\Pi^\star_{\mathrm{F}}$ changes to:

$$\Pi^\star_{\mathrm{F}}(\Box\Phi, u) = \forall v{:}\, W\ \Pi^\star_{\mathrm{F}}(\Phi, v)$$

where the other cases remain as in Definition 2.2.
Although it might not be very surprising, the simplicity of how this approach allows to reason within first-order S5 is striking.

5.1.6 The Modal Logic KDE (or KD5)

Working with KDE is almost as easy as with S5, since euclidity is very closely related to equivalence. Under the assumption that the frames we consider are R-connected (and this can be assumed w.l.o.g.), it guarantees that any two worlds which both are not identical to the actual world, have access to each other. In fact this holds for any two worlds which are accessed somehow. Thus a suitable unit-clause to be added is simply $R(ux, vy)$ instead of the more complicated $R(v, w) \leftarrow R(u, v), R(u, w)$. Furthermore this means that the unit-clause $R(u, ux)$ which usually has to be included for any serial modal logic, can be simplified to $R(\iota, x)$. Both unit clauses $R(ux, vy)$ and $R(\iota, x)$ do not interfere since the latter can can only be used for an R-query with ι as the first argument and the former only for R-queries where ι is *not* the first argument. This allows to incorporate the two theory clauses directly in the translation by:

$$\begin{array}{l} \Pi^\star_{\mathrm{F}}(\Box\Phi, \iota) = \forall x{:}\, F_R\ \Pi^\star_{\mathrm{F}}(\Phi, [x]) \\ \Pi^\star_{\mathrm{F}}(\Box\Phi, u) = \forall v{:}\, W\ \forall x{:}\, F_R\ \Pi^\star_{\mathrm{F}}(\Phi, [vx]) \\ \qquad\qquad\qquad\qquad\qquad\qquad \text{if } u \neq \iota \\ \text{The other cases remain unchanged} \end{array}$$

[2]Two worlds u and v are R-connected iff there exist $x_1, \ldots, x_n$ with $R(x_i, x_{i+1})$ or $R(x_{i+1}, x_i)$ and $u = x_1$ and $v = x_n$, i.e. they belong to the same R-equivalence class.

This shows that reasoning within KDE is almost as easy as within S5.

5.1.7 The Modal Logic KD4E (or KD45)

The only difference between KDE and KD4E lies in the fact that in KD4E any world but the actual world ι can be accessed by ι which is not necessarily true for KDE. This can be utilized by changing the unit-clause $R(ux, vy)$ to $R(u, vy)$, i.e. ι is the only world which possibly cannot be accessed. Obviously $R(u, vy)$ subsumes $R(u, ux)$ therefore even $R(u, ux)$ can be omitted.
Here again we are in the fortunate situation that the background theory of the accessibility relation can be described by a single unit clause. Hence its effect can also be put directly into the formula morphism which thus changes to:

$$\Pi^\star_{\mathrm{F}}(\Box\Phi, u) = \forall v{:}\, W\, \forall x{:}\, F_R\, \Pi^\star_{\mathrm{F}}(\Phi, vx)$$

and the other cases remain as in Definition 2.2.
Note that for both, KDE and KD4E, the additional assumption that ι has access to itself (or even is accessed somehow) leads to S5.

5.1.8 The Modal Logic KT (or T or M)

This modal logic is characterized by the reflexive frames. Thus the additional theory axioms are simply:

$$R(u, ux).$$
$$R(u, u).$$

Unfortunately these two unit clauses do not allow any further simplification as it was possible in the previous cases. Nevertheless they behave slightly better than pure relational translation because, firstly, the extra clauses which contain nothing but R-literals are avoided and, secondly, $\Diamond$-formulae get smaller.

5.2 Non-Serial Modal Logics

If seriality is not assumed the additional "normality"-predicate introduced by the formula morphism must not be ignored. This means that the simplifications introduced above have to take care of the additional N-symbol because they will be introduced as well by the (unsimplified) clause $R(u, ux) \leftarrow N(u)$. Therefore the transitivity clause $R(u, vx) \leftarrow R(u, v)$ becomes $R(u, vx) \leftarrow N(v), R(u, v)$, the symmetry clause $R(ux, u)$ becomes $R(ux, u) \leftarrow N(u)$ and the euclidity clause $R(ux, vy)$ becomes $R(ux, vy) \leftarrow N(u), N(v)$. Thus the additional PROLOG clauses to be added are:

$R(u, ux)$	$\leftarrow$	$N(u)$	for	K
$R(ux, u)$	$\leftarrow$	$N(u)$	for	B
$R(u, vx)$	$\leftarrow$	$N(v), R(u, v)$	for	4
$R(ux, vy)$	$\leftarrow$	$N(u), N(v)$	for	E (or 5)
$R(u, vy)$	$\leftarrow$	$N(v)$	for	K4E (or K45)

Some problems might arise in the non-serial case because there is no such nice result as in Lemma 5.2. I.e. whether or not a formula is Horn after translation can't be found out by simply looking at its "flattened" form. Getting similar results for non-seriality is a matter for future work.

Chapter 6

Varying Domains

In varying domains we do no more assume that there is a single domain common to all worlds, i.e. each world might have its own universe of discourse. This slightly changes some of the former definitions from Chapter 1. For instance, interpretations do not refer to a unique domain, but the respective domains are all given by the mapping $\Im_{\mathrm{loc}}$ which maps worlds to structures. Also the satisfiability relation changes accordingly:

Definition 6.1 (Satisfiability in Varying Domains)
Let $\Im_{\mathrm{M}} = (\mathcal{D}, \mathcal{F}_{\mathrm{M}}, \Im_{\mathrm{loc}}, \tau, \phi)$ be a modal logic interpretation.

$$\begin{array}{lll} \Im_{\mathrm{M}} \models_{\mathrm{M}} \forall x{:}\, D\ \Phi & \text{iff} & \Im_{\mathrm{M}}[x/a] \models_{\mathrm{M}} \Phi \text{ for every } a \text{ in the domain of } \Im_{\mathrm{loc}}(\tau) \\ \Im_{\mathrm{M}} \models_{\mathrm{M}} \exists x{:}\, D\ \Phi & \text{iff} & \Im_{\mathrm{M}}[x/a] \models_{\mathrm{M}} \Phi \text{ for some } a \text{ in the domain of } \Im_{\mathrm{loc}}(\tau) \end{array}$$

The other cases remain as in Definition 1.7

This new definition obviously has its immediate effect on the relational translation of modal logic formulae. In addition to the new predicate symbol R we also have to add a symbol E which is supposed to represent the "existence" relation.

Definition 6.2 (The Formula Morphism)

$$\begin{array}{lll} \Pi_{\mathrm{F}}(\forall x{:}\, D\ \Phi, u) & = & \forall x{:}\, D\ E(u,x) \Rightarrow \Pi_{\mathrm{F}}(\Phi, u) \\ \Pi_{\mathrm{F}}(\exists x{:}\, D\ \Phi, u) & = & \exists x{:}\, D\ E(u,x) \wedge \Pi_{\mathrm{F}}(\Phi, u) \end{array}$$

All other cases remain as in Definition 2.2.

Note that by this definition it is easy to see that constant domains are a special case of varying domains since in constant domains the additional unit clause $E(u,x)$ which just states that any element occurs in every world simplifies the above definition to Definition 2.2.

We omit the soundness and completeness proofs of this formula translation since they do not provide anything particularly new.

More interesting is a closer look at the E-predicate and where it occurs inside translated formulae. As a matter of fact, any occurrence of it is very similar to the occurrences of R-predicates inside translated formulae. Thus, the introduction of functional simulators for this "existence" predicate allows for a considerable simplification as follows:

Lemma 6.3
Let F_E be a functional simulator for the binary relation E. Then each element of F_E denotes a total function and the relational translation can be modified to:

$$\begin{aligned}\Pi^\star_{\mathrm{F}}(\forall x{:}\, D\ \Phi, u) &= \forall y{:}\, F_E\ \Pi_{\mathrm{F}}(\Phi, u)[x/uy] \\ \Pi^\star_{\mathrm{F}}(\exists x{:}\, D\ \Phi, u) &= \exists y{:}\, F_E\ \Pi_{\mathrm{F}}(\Phi, u)[x/uy]\end{aligned}$$

where $\Pi_{\mathrm{F}}(\Phi, u)[x/uy]$ means that any occurrence of x inside $\Pi_{\mathrm{F}}(\Phi, u)$ is to be replaced by $[u\, y]$.
Proof: Totality of the functions is guaranteed by the fact that the domain of any world is not empty.
Soundness and completeness of the translation morphism $\Pi^\star_{\mathrm{F}}$ follows similarly to Theorem 3.8 and Section 5.1.1.

Note that no further simulator axiom is to be included.

This translation already takes into account that there is no further theory behind the "existence"-predicate. Very often, however, one is interested in certain extra properties as e.g. increasing domains or decreasing domains. These are handled in the following two sections.

6.1 Increasing Domains

The characteristic axiom schema for increasing domains is:

$$\forall u, v{:}\, W\ \forall x{:}\, D\ E(u, x) \wedge R(u, v) \Rightarrow R(v, x)$$

i.e. if the domain element x exists in world u und v is accessible from u then x exists in v as well. W.l.o.g. we assume that the accessibility relation is not symmetric because this would immediately force a constant domain structure which has been handled already before.
In arbitrary varying domains the only theory clause to be included was the simple unit clause $\forall u{:}\, W\, \forall x{:}\, F_E\ \ E(u, u\, x)$ and in fact, we were able to work this up by modifying the formula morphism just as in the case for the logic KD.
For more complicated modal logics the additional theory axioms could not be simplified to a single unit clause and therefore a translation had to be chosen which allows the occurrence of negative R-literals, but nevertheless also allowed a simplification

of the theory axioms. Similar things happen here for increasing domains. First of all, the translation morphism from above has to be changed to:

$$\begin{array}{rcl} \Pi^\star_{\mathrm{F}}(\forall x\colon D\ \Phi, u) & = & \forall x\colon D\ E(u,x) \Rightarrow \Pi_{\mathrm{F}}(\Phi, u) \\ \Pi^\star_{\mathrm{F}}(\exists x\colon D\ \Phi, u) & = & \exists y\colon F_E\ \Pi_{\mathrm{F}}(\Phi, u)[x/uy] \end{array}$$

i.e. $\Pi^\star_{\mathrm{F}}$ behaves on $\forall$-formulae like it does on $\Box$-formulae. It is easy to see that the clause normal form of the translation of any modal logic formula which is in negation normal form contains no positive E-literal.
Now we are in a position where we can try to simplify the theory axiom for increasing domains, again analogously to the simplification of accessibility relation properties given earlier.
A first simplification is:

$$\forall u\colon W\ \forall x\colon F_R\ \forall y\colon D\ E(ux, y) \leftarrow E(u, y)$$

if R is serial or

$$\forall u\colon W\ \forall x\colon F_R\ \forall y\colon D\ E(ux, y) \leftarrow N(u), E(u, y)$$

if seriality of R is not assumed.
It can easily be checked that the additional assumption of R's transitivity does not lead to anything futher. Also reflexivity has no effect not yet stated by the above simplification. Therefore this simplification is indeed characteristic for increasing domains if we consider the modal logics K, KD, KT, K4, KD4, and S4[1].

Euclidity of the accessibility relation also doesn't lead to particular problems. Note again, that frames which obey euclidity either form an equivalence closure (a non-degenerated cluster as it was called in [9]) or a single world followed by such an equivalence closure. Obviously there are constant domains inside clusters. Thus the only additional axioms to be added are those which state that any domain element occurs in any world different from the initial world ι, and this can simply be done by the unit clause $\forall u\colon W\,\forall x\colon F_R\,\forall y\colon F_E\ E(ux, vy)$. This even allows to simplify the simulator axiom $E(u, ux)$ to $E(\iota, [x])$ and indeed these two unit clauses suffice for both, KD5 and KD45.
Adding reflexivity or symmetry immediately leads to constant domains and the non-serial cases are captured by the slightly modified clause

$$\forall u, v\colon W\,\forall x\colon F_R\,\forall y\colon F_E\ E(ux, vy) \leftarrow N(u)$$

6.2 Decreasing Domains

The characteristic axiom schema for decreasing domains is:

$$\forall u, v\colon W\ \forall x\colon D\ E(u, x) \wedge R(v, u) \Rightarrow R(v, x)$$

[1]Note that the simplified axiom also guarantees that domain elements exist in worlds which can be accessed by more than one R-step, regardless of whether R is transitive or not.

Its similarity to the axiom schema for increasing domains suggests an analogous way of dealing with it. The reader may verify by himself that the theory clauses to be added then are $E(u, u\, y_E)$ and $E(u, z) \leftarrow E(u\, x_R, z)$ for KD, KT, KD4, and KT4 and $E(\iota, [x_E])$ and $E(u, v\, x_R\, y_E)$ for KD5 and KD45. It is evident that for the non-serial modal logics the respective N-literals have to be taken into account.

Chapter 7

Conclusion

In this paper an approach is presented which allows to reason within modal logics by some appropriate translation into first-order predicate logic. One of the aims of this development was to avoid theory unification and/or equality handling (see [7] and [8]) such that even an inference machine like PROLOG can be utilized as a theorem prover for modal logics. For two main reasons this translation had to be different to the one proposed by Moore in [6]. First of all, Moore's approach (which is essentially the relational translation from chapter 2) produces formulae which contain lots and lots of literals (clauses) which merely express features of the accessibility relation such that proofs of even small theorems become fairly complicated and lengthy if at all feasible. The other reason has to do with the PROLOG control. As it was shown in some examples, the purely relational translation may produce formulae which (although they are in Horn form) do not terminate when interpreted by PROLOG. This holds in particular for properties like "Symmetry" and "Transitivity". It is one of the big advantages of the approach proposed in this report that such loops are impossible.

The main idea behind the method presented here is the *functional simulation* of the accessibility relation by a suitable set of functions. This turned out to have two main advantages: it decreases the size and the number of clauses and it allows significant simplifications on the set of theory clauses. These simplifications were the key to the utilization of PROLOG as an inference machine for modal logics as they avoid infinite loops which are inevitable for many cases in the pure relational translation approach.

The work on functional simulation of accessibility relations is still in the beginning. There are certainly more possible applications to modal logics which could not be handled in this paper. A short collection of some immediate ideas follows in the next chapter.

Chapter 8

Further Work

The whole approach seems strong and flexible enough to deal with much more cases than introduced here. So, for example, we can think of further modalities, i.e. further accessibility relation properties not mentioned above. Another possibility is the examination of multi-modalities as it is very common in the area of multi-agent systems. Nevertheless there certainly are some limitations of this method and these should be mentioned as well, as far as they are known up to now.

8.1 Other Accessibility Relation Properties

The properties considered so far are actually the most common ones occurring in the standard literature. At the time being we haven't examined much more possible properties in detail. So e.g. another quite interesting property is *density*. An accessibility relation R is called *dense* if for any two worlds u and v with $R(u, v)$ there is a world w such that both $R(u, w)$ and $R(w, v)$ hold. Obviously this property is only interesting in frames where reflexivity is not assumed because otherwise such a possible candidate can trivially be found. The additional clauses should thus be:

$$\begin{array}{rcl} R(u, f(u, v)) & \leftarrow & R(u, v). \\ R(f(u, v), v) & \leftarrow & R(u, v). \end{array}$$

It is not easy to see how these two clauses can be simplified according to the techniques we introduced for the various modal logics. Nevertheless there is a simple unit clause which seems to cover all the responsibilities of the two clauses from above, namely $R(u\, f(u, ux), ux)$ which states that for any world u and any R-successor ux of u there is a world accessible from u which can access ux. Whether this is indeed a characteristic axiom for density has not been examined in detail up to now and is left as an open question.

Example 8.1
Prove: $\Box\Diamond P \Rightarrow \Box\Diamond\Diamond P$ in KD(dense):

$$\begin{array}{lcl}
P(ua) & \leftarrow & R(\iota, u) \\
R(u, ux). & & \\
R(uf(u, ux), ux). & & \\
 & \leftarrow & P(w), R(b, v), R(v, w) \\
 & \leftarrow & R(\iota, u), R(b, v), R(v, ua) \\
 & \leftarrow & R(b, v), R(v, xa) \\
 & \leftarrow & R(by, xa) \\
 & \leftarrow & \text{yes}
\end{array}$$

Another pretty interesting property which frequently occurs in the literature is *directedness*. An accessibility relation R is *directed* if for any worlds u, v and w with $R(u, v)$ and $R(u, w)$ there exists a world which is accessible by both v and w. This forces a sort of confluence on the accessibility relation.
This property can be expressed by two clauses, namely:

$$\begin{array}{lcl}
R(v, f(u, v, w)) & \leftarrow & R(u, v), R(u, w) \\
R(w, f(u, v, w)) & \leftarrow & R(u, v), R(u, w)
\end{array}$$

Again it seems not easy to simplify these two clauses. However, if we consider the unit clause $R(u\,x, u\,y\,f(u, ux, uy))$ it looks as if it covers all the possibilities of the two unsimplified clauses since it expresses that for any two worlds which have a common predecessor there exists a world accessible from the second one which can be accessed by the first one as well. Proving that this unit clause is indeed characteristic for directed world structures is again a matter of future work.

Example 8.2
Prove: $\Diamond\Box P \Rightarrow \Box\Diamond P$ in KD(directed):

$$\begin{array}{lcl}
P(u) & \leftarrow & R(a, u) \\
R(u, ux). & & \\
R(u\,x, u\,y\,f(u, ux, uy)). & & \\
 & \leftarrow & P(v), R(b, v) \\
 & \leftarrow & R(a, u), R(b, u) \\
 & \leftarrow & R(b, ax) \\
 & \leftarrow & \text{yes}
\end{array}$$

In transitive frames an even simpler version of the directedness axiom seems sufficient, namely $R(u, v\,f(u, v))$, i.e. for any two worlds u and v there is a world accessible from v which is also accessible from u. It is not explicitly required that u and v share a common predecessor because this is guaranteed by the transitivity property anyway.

Although it hasn't been checked in detail up to now, a suitable axiomatization of the modal logic S4.2 thus simply contains the three unit clauses:

$$\begin{array}{l} R(u,[u\,x]) \\ R(u,u) \\ R(u,[v\,f(u,v)]) \end{array}$$

Finally, let us consider accessibility relations which are partially functional. This property can be expressed by the axiom schema

$$\Diamond\Phi \Rightarrow \Box\Phi$$

i.e. if some world can be accessed then this world is the only accessible one.
The corresponding axiom is thus:

$$\forall u,v,w\; R(u,v) \wedge R(u,w) \Rightarrow v = w$$

Two resolution steps with the unit clause $R(u,ux)$ result in the equation $ux = uy$. We cannot put this equation as an additional clause to our Horn formulae because standard PROLOG wouldn't know how to deal with it. However, we *can* have a closer look at what this equation can possibly cause. There is actually only one case where this equation can come into play and that is by replacing any world-path prefix $[\alpha_1 \ldots \alpha_n \beta]$ by $[\alpha_1 \ldots \alpha_n y]$. This effect can be put already into the translation such that the $\Pi^\star_\mathrm{F}$ can be changed to

$$\Pi^\star_\mathrm{F}(\Diamond\Phi, u) = \forall y{:}\, F_R\; \Pi^\star_\mathrm{F}(\Phi, uy)$$

All the other cases remain as in Definition 3.5.

Example 8.3
Prove: $\Diamond\Diamond P \Rightarrow \Diamond\Box P$ in KD(functional)
Negation and transformation into clause normal form results in:

$$\begin{array}{ll} P([x\,y]). & \\ & \leftarrow\quad P([z_1\,z_2]) \end{array}$$

PROLOG evidently has no problems with this.

8.2 Multi-Modalities

As an example let us consider an epistemic logic for multiple agents. The most common modal logic for representing believes is K45 (or KD45), i.e. it is assumed that the agent has full introspection, i.e. s/he knows what s/he knows and s/he knows what s/he doesn't know[1]. For each agent a assume an accessibility relation (epistemic alternative relation) R_a. Now recall that the additional theory clause to be introduced for K45 was $R(u, vx) \leftarrow N(v)$ which states that any world not equal to the initial one (if there is one) can be accessed from everywhere. Something similar has to happen for multiple K45-modalities. However, we actually do not want to express that for every agent a $R_a(u, vx_a) \leftarrow N_a(v)$ holds for arbitrary u and v but only for those which are R_a-connected to ι. I.e. we want to be sure that both u and v can be accessed by agent a starting from ι. If there is only one modality then this is indeed equivalent to $R_a(u, vx_a) \leftarrow N_a(v)$. Therefore we have to introduce further predicates which are supposed to express that certain worlds belong to the epistemic alternatives of certain agents. Let us use the symbol W indexed with the agents identifier to denote these predicates. Thus the theory clause becomes now:

$$\forall u, v\, \forall x\colon \text{Agents}\, \forall y\colon F_{R_x}\ R_x(u, vy) \leftarrow N_x(v), W_x(u), W_x(v)$$

Certainly this is not yet enough; it still has to be expressed when $W_x(u)$ is true for some agent x and some world denoted by the world-path u. However, this is fairly simple: u is an x-accessible world if either u is just the initial world ι or it is a world which is accessed from some x-accessible world by a simulator function for x[2]. Summarizing we get the following translation rules

$$\begin{array}{rcl}\Pi^\star_{\mathrm{F}}(\Box_{\text{agent}}\Phi, u) & = & \forall v\colon W\ R_{\text{agent}}(u, v) \Rightarrow \Pi^\star_{\mathrm{F}}(\Phi, v)\\ \Pi^\star_{\mathrm{F}}(\Diamond_{\text{agent}}\Phi, u) & = & N_{\text{agent}}(u) \wedge \exists v\colon W\ \Pi^\star_{\mathrm{F}}(\Phi, uv)\end{array}$$

and the following theory clauses

$$\begin{array}{lcl}R_x(u, uy_x) & \leftarrow & N_x(u).\\ R_x(u, vy_x) & \leftarrow & N_x(v), W_x(u), W_x(v).\\ W_x(\iota). & & \\ W_x([v\, y_x]) & \leftarrow & W_x(v).\end{array}$$

Note that the clause $R_x(u, uy_x) \leftarrow N_x(u)$ has to be included again because it is no longer subsumed by other theory clauses.

If we consider the modal logic KD45 as the one which suits to epistemic logic (i.e. we assume that the agent's belief is consistent) it is worth noting that the translation from above does not change. The reason is that although the accessibility relation of each agent might be serial it is not necessarily true that all the other agents know about this. It is nowhere stated in the above translation morphism definition

[1]There is still an argument on whether it is really appropriate to take any logic based on K as an epistemic logic. Also it is questionable to allow full introspection. I believe that the reason behind this argument is that we usually assume a very strong connection between "knowing" something and "being aware of" something. If we represent Knowledge and Belief by KD45 we essentially handle them as 'knowing in principle; s/he could know it, if s/he just thought about it long enough.

[2]This certainly will look slightly different in the actual PROLOG implementation. For instance, W would be represented by some binary predicate whose first argument is an agent and second argument is a world-path (or actually a world-term using "apply"-functions).

that the world u is indeed an epistemic alternative of the agent under consideration. Thus the additional normality predicate has to be kept.
However, the theory clauses do change a bit. First of all there is another possibility for a world to be normal: it just may be x-accessible. But this simplifies the second theory clause from above and we finally get

$$\begin{array}{lcl} R_x(u, uy_x) & \leftarrow & N_x(u). \\ R_x(u, vy_x) & \leftarrow & W_x(u), W_x(v). \\ W_x(\iota). & & \\ W_x([v\, y_x]) & \leftarrow & W_x(v). \\ N_x(u) & \leftarrow & W_x(u). \end{array}$$

This way, we have the possibility to reason within a multi-agent scenario where each of the agents' believes works according to the modal logic K45 (or KD45).

As it stands, the respective belief spaces are entirely independent. Nevertheless, one usually wants to have a possibility to express something like a mutual belief of a set of agents. A formula Φ is *mutually believed* if every agent believes Φ and every agent believes that every other agent (and he himself) believes Φ and so on for arbitrary long belief sequences. Thus mutual belief is a further modality although very closely related to the individual belief modalities. Let us examine the properties it obeys: First of all we certainly cannot assume reflexivity, for what is mutually believed is not necessarily true in the reality (the whole group of agents might be mistaken). On the other hand, from the consistency assumption of the individual agents's believes it follows directly that mutual belief has to be consistent as well. Similarly, transitivity and euclidity of the individual accessibility relations directly induces both transitivity and euclidity to mutual belief. Thus we have to choose a K45 (KD45) modality for mutual belief as well. Let us call the mutual belief accessibility relation R_{MB}. Then we have for any two arbitrary worlds u and v that $R_{\mathrm{MB}}(u, v\, y_{\mathrm{MB}})$ because by definition every world (but ι) is accessible by R_{MB}. This does unfortunately not yet work perfectly together with the individual simulator functions. The problem is that mutual belief subsumes individual belief but this cannot be proved from the above since it is not obvious for the inference machine that it is allowed to instantiate mutual belief simulator function variables with individual belief simulator function symbols. Fortunately this problem can be solved quite easily: we just have to change $R_{\mathrm{MB}}(u, v\, y_{\mathrm{MB}})$ to

$$R_{\mathrm{MB}}(u, v\, y_x)$$

where x ranges over the set of agents for which the mutual belief holds. With this we can now easily prove that mutual belief subsumes individual belief.

Example 8.4
Prove: $\Box_{\mathrm{MB}} P \Rightarrow \Box_{\mathrm{John}} P$ in multiple KD45 (and mutual belief)

The clause form we get after translation is (together with the derivation):

$$\begin{array}{lll}
P(u) & \leftarrow & R_{\mathrm{MB}}(\iota, u). \\
N_{\mathrm{John}}(\iota). & & \\
R_{\mathrm{MB}}(u, vy_x). & & \\
\multicolumn{3}{l}{\text{the other theory clauses are not necessary}} \\
 & \leftarrow & P(a_{\mathrm{John}}) \\
 & \leftarrow & R_{\mathrm{MB}}(\iota, a_{\mathrm{John}}) \\
 & \leftarrow & \text{yes}
\end{array}$$

It thus takes two simple inference steps to prove the theorem.

8.3 Restrictions

A necessary condition for the method proposed here to work is that the accessibility relation properties have to be first-order predicate logic definable. Unfortunately not all modal logics have a first-order describable frame property. As an example consider the following axiom schema:

$$\Diamond\Phi \wedge \Box(\Phi \Rightarrow \Diamond\Phi) \Rightarrow \Box\Diamond\Phi$$

It represents the *discreteness* of the underlying accessibility relation, i.e. it guarantees that for any two worlds there are only finitely many in between. This cannot be expressed in first-order predicate logic; therefore there is no suitable correspondence axiom to be included and our approach won't work.
Similarly there is no first-order correspodence axiom for the schema:

$$\Box\Diamond\Phi \Rightarrow \Diamond\Box\Phi$$

which is known as the McKinsey axiom. The correspondence formula for this schema would indeed require a so-called Henkin-quantifier (or "branching" quantifier) which can't be expressed in first-order predicate logic.

But there is a further restriction if we insist on using PROLOG as an inference machine: the correspondence axiom might not be Horn. One example of such a non-Horn property is *linearity.* In a linear frame any two worlds are comparable, i.e. for any u and v: either they are identical or v is accessible from u or u is accessible from v. This is a typical non-Horn formula and hence causes problems for PROLOG. Nevertheless, most accessibility relation properties one is usually interested in *are* Horn and this makes the whole approach feasible for many applications.

Another possibility for some undesired behaviour of the method lies again in the particular way PROLOG is dealing with clauses and with literals inside clauses. Recall that the predicate logic formulae we get after translation look very similar to their respective flattened forms (i.e. the formulae we would get if we ignored the modal operators at all). PROLOG tends to run into loops whenever there are modal logic formulae whose flattened form is a predicate logic tautology. For instance consider the formula $\Box P \vee \Box \neg P$. Its flattened form is just $P \vee \neg P$ which is a tautology. Its translated version is (in PROLOG notation): $P(u) \leftarrow R(\iota, u), R(\iota, v), P(v)$, i.e. P is true in any accessible world if it is true in some accessible world. Such a formula typically occurs when *knowing whether* is to be expressed because it is "known whether" P is true if and only if either P is known to be true or $\neg P$ is known to be true. It is evident that a Horn clause like this one is a hot candidate for being a cause of infinite loops in PROLOG. However, it does not necessarily produce loops as can be seen in the example in the following chapter.

Chapter 9

Example: the red hat puzzle

Assume three people, Annie, Bernard and Charlotte, seated in a row such that Annie sees both Bernard and Charlotte, Bernard sees Charlotte and Charlotte (the unhappy one) isn't able to see anyone of the others. Each of them wears a hat but none of them knows the colour of his own hat. However, they are told that at least one of the hats is red. Then they are asked whether there is one of them who knows something about the colour of his hat. Annie is the first who answers and she says: "Unfortunately, I don't know whether my hat is red." Then Bernard says: "Neither do I, I'm afraid." Finally Charlotte raises a hand and says: "Well, but I know. Evidently I am wearing a red hat!" How does she know?

Let us try to solve this puzzle with the help of the approach presented in this paper. First of all we have to represent the given information logically. I.e. we have to express that both Annie and Bernard know whether Charlotte has got a red hat and that in addition Annie also knows whether Bernard's hat is red. "Knowing whether" can be represented by "either knowing that . . . or knowing that not . . . ". We therefore get the following formulae:

$\Box_A \operatorname{red}(B) \vee \Box_A \neg\operatorname{red}(B)$	for: Annie knows whether Bernard's hat is red
$\Box_A \operatorname{red}(C) \vee \Box_A \neg\operatorname{red}(C)$	for: Annie knows whether Charlotte's hat is red
$\Box_B \operatorname{red}(C) \vee \Box_B \neg\operatorname{red}(C)$	for: Bernard knows whether Charlotte's hat is red

The next information we got is that at least one of the hat's is red, which is represented by:

$$\operatorname{red}(A) \vee \operatorname{red}(B) \vee \operatorname{red}(C)$$

Finally, we know that neither Annie nor Bernard know whether their respective hats are red:

$$\neg(\Box_A \operatorname{red}(A) \vee \Box_A \neg\operatorname{red}(A))$$
$$\neg(\Box_B \operatorname{red}(B) \vee \Box_A \neg\operatorname{red}(B))$$

These are essentially the facts given in the puzzle. It might be not too surprising, however, that this is not yet enough in order to derive the desired solution. For instance, although these are the given facts, it is not said anywhere yet that the three candidates indeed know that these are the facts. And even this wouldn't be sufficient because it also has to expressed that everyone knows that each of the others knows the facts and so on. In other words, any of the above formulae has to occur in the context of a mutual belief operator[1]. Thus we finally get the follwing set of formulae:

$$\begin{array}{l}\Box_{\mathrm{MB}}(\Box_A \mathrm{red}(B) \vee \Box_A \neg\mathrm{red}(B))\\ \Box_{\mathrm{MB}}(\Box_A \mathrm{red}(C) \vee \Box_A \neg\mathrm{red}(C))\\ \Box_{\mathrm{MB}}(\Box_B \mathrm{red}(C) \vee \Box_B \neg\mathrm{red}(C))\\ \Box_{\mathrm{MB}}(\neg(\Box_A \mathrm{red}(A) \vee \Box_A \neg\mathrm{red}(A)))\\ \Box_{\mathrm{MB}}(\neg(\Box_B \mathrm{red}(B) \vee \Box_A \neg\mathrm{red}(B)))\end{array}$$

and the theorem

$$\Box_C \mathrm{red}(C)$$

which states that Charlotte knows that her hat is red. Now, these six formulae have to be translated with the help of our formula morphism and we get after transformation into clause normal form (without theory axioms):

$$\begin{array}{ll}\neg R_{\mathrm{MB}}(\iota, u) \vee \neg R_A(u, v) \vee \mathrm{red}(v, B) \vee \neg R_A(u, w) \vee \neg\mathrm{red}(v, B) & \\ \neg R_{\mathrm{MB}}(\iota, u) \vee \neg R_A(u, v) \vee \mathrm{red}(v, C) \vee \neg R_A(u, w) \vee \neg\mathrm{red}(v, C) & \\ \neg R_{\mathrm{MB}}(\iota, u) \vee \neg R_B(u, v) \vee \mathrm{red}(v, C) \vee \neg R_B(u, w) \vee \neg\mathrm{red}(v, C) & \\ \neg R_{\mathrm{MB}}(\iota, u) \vee \mathrm{red}(u, A) \vee \mathrm{red}(u, B) \vee \mathrm{red}(u, C) & \\ \neg R_{\mathrm{MB}}(\iota, u) \vee \neg\mathrm{red}([u\, a_A(u)], A) & \\ \neg R_{\mathrm{MB}}(\iota, u) \vee \mathrm{red}([u\, b_A(u)], A) & \\ \neg R_{\mathrm{MB}}(\iota, u) \vee \neg\mathrm{red}([u\, c_B(u)], B) & \\ \neg R_{\mathrm{MB}}(\iota, u) \vee \mathrm{red}([u\, d_B(u)], B) & \\ \neg\mathrm{red}([e_C], C) \qquad \text{the negated theorem} & \end{array}$$

The first problem already arises: there is a non-Horn clause, namely the fourth one. However this problem is not too hard to be solved. The above clause set can be made Horn if we just replace any occurrence of red by $\neg r$ and, of course, any occurence of $\neg$red by r[2]. Hence, what remains to be done is that we have to agree on the properties of the individual belief operators. For simplicity we do not assume anything particular but consistency (seriality). This is not even necessary but it simplifies the derivation. We therefore assume that the consistency of the respective agent's knowledge is mutually believed and this can be represented by the simple unit clause: $R_x(u, u\, y_x)$.
The mutual belief accessibility relation is transitive over the individual believes and we thus get the two clauses:

$$\begin{array}{l}R_{\mathrm{MB}}(u, u\, y_x)\\ R_{\mathrm{MB}}(u, v\, y_x) \leftarrow R_{\mathrm{MB}}(u, v)\end{array}$$

[1]Note that, if we (humans) solve this puzzle, we implicitly assume such a mutual belief without really thinking about it.

[2]This is a usual way which fairly often works to transform non-Horn clauses into Horn clauses.

As long as there is no explicit agent's reasoning about mutual belief (i.e. the first argument of the relation is always ι) this can even be simplified to

$$R_{\text{MB}}(\iota, v\, y_x)$$

But still we are not ready yet. From the above it follows quite easily that e.g. Annie believes that either Bernard or Charlotte has a red hat. It is even so that Bernard can derive this. However, we have not expressed that Bernard trusts in Annie's believes, not even what the colour of the other's hats is concerned. Nevertheless, this can be fixed easily if we assume that the correctness of the individual believes on the other's hat colours is mutually believed, i.e. $\Box_{\text{MB}}(\Box_x red(y) \Rightarrow red(y))^3$.

We are now almost done with the necessary preparations; the above clause set can be given to PROLOG and we can try to run it as a program. But, evidently, we do so after arranging the clauses in a way such that the potential candidates for infinite loops come last and we get the following final Horn clause set:

$$\begin{array}{lcl}
r([u\, a_A(u)], A) & \leftarrow & R_{\text{MB}}(\iota, u). \\
r([u\, c_B(u)], B) & \leftarrow & R_{\text{MB}}(\iota, u). \\
r([e_C], C). & & \\
r([u\, k_x(u)], y) & \leftarrow & r(u, y), R_{\text{MB}}(\iota, u). \\
r(v, B) & \leftarrow & R_A(u, v), R_A(u, w), R_{\text{MB}}(\iota, u), r(w, B). \\
r(v, C) & \leftarrow & R_A(u, v), R_A(u, w), R_{\text{MB}}(\iota, u), r(w, C). \\
r(v, C) & \leftarrow & R_B(u, v), R_B(u, w), R_{\text{MB}}(\iota, u), r(w, C). \\
R_{\text{MB}}(\iota, [v\, y_x]). & & \\
R_x(u, [u\, y_x]). & & \\
 & \leftarrow & r([u\, b_A(u)], A), R_{\text{MB}}(\iota, u). \\
 & \leftarrow & r([u\, d_B(u)], B), R_{\text{MB}}(\iota, u). \\
 & \leftarrow & r(u, A), r(u, B), r(u, C), R_{\text{MB}}(\iota, u).
\end{array}$$

Modulo the actual implementation of the world paths this is a clause set which solves the puzzle. The readers which are interested in the "real" PROLOG program might try to run the following "pure" PROLOG code:

World paths like $[u\, \alpha_x \dots \beta_y]$ are represented by terms as:

$$app(s(\beta, y), app(\dots, app(s(\alpha, x), u)))$$

where the function symbol $s(u, v)$ is used to denote that u is a belief function for agent v, written earlier as u_v. "app" obviously is the application function and "rel(agent,u,v)" means that world v is accessible from world u by the relation R_{agent}, i.e. $R_{\text{agent}}(u, v)$. Finally, "initial" denotes the initial world ι and rMB denotes the mutual belief accessibility relation.

[3] We could also use "Knowledge" instead of "Belief" with the difference that something is known if it is belived and true.

```
r(app(s(a(U),annie),U),annie) :- rMB(initial,U).
r(app(s(c(U),bernard),U),bernard) :- rMB(initial,U).
r(app(s(e,charlotte),initial),charlotte).
r(app(s(k(U),X),U),Y) :- r(U,Y),rMB(initial,U).
r(V,bernard) :- rel(annie,U,V),rel(annie,U,W),rMB(initial,U),r(W,bernard).
r(V,charlotte) :- rel(annie,U,V),rel(annie,U,W),rMB(initial,U),r(W,charlotte).
r(V,charlotte) :- rel(bernard,U,V),rel(bernard,U,W),rMB(initial,U),r(W,charlotte).

rMB(initial,app(s(Y,X),V)).

rel(X,U,app(s(Y,X),U)).

:- r(app(s(b(U),annie),U),annie),rMB(initial,U).
:- r(app(s(d(U),bernard),U),bernard),rMB(initial,U).
:- r(U,annie),r(U,bernard),r(U,charlotte),rMB(initial,U).
```

It might be interesting to have a look at the proof PROLOG produces and to translate it back into modal logic.

given	$\Box_{\mathrm{MB}}(red(Annie) \vee red(Bernard) \vee red(Charlotte))$
implies	$\Box_{\mathrm{MB}}\Box_A(red(Annie) \vee red(Bernard) \vee red(Charlotte))$
given	$\Box_{\mathrm{MB}}\Diamond_A\neg red(Annie)$
	by $\Box_{\mathrm{MB}}\neg(\Box_A red(Annie) \vee \Box_A\neg red(Annie))$
resolvent	$\Box_{\mathrm{MB}}\Diamond_A(red(Bernard) \vee red(Charlotte))$
given	$\Box_{\mathrm{MB}}(\Box_A red(Bernard) \vee \Box_A\neg red(Bernard))$
resolvent	$\Box_{\mathrm{MB}}(\Box_A red(Bernard) \vee \Diamond_A red(Charlotte))$
given	$\Box_{\mathrm{MB}}(\Diamond_A\neg red(Bernard) \vee red(Bernard))$
	by $\Box_{\mathrm{MB}}(\Box_x rY \Rightarrow rY)$
resolvent	$\Box_{\mathrm{MB}}(red(Bernard) \vee \Diamond_A red(Charlotte))$
implies	$\Box_{\mathrm{MB}}\Box_B(red(Bernard) \vee \Diamond_A red(Charlotte))$
given	$\Box_{\mathrm{MB}}\Diamond_B\neg red(Bernard)$
	by $\Box_{\mathrm{MB}}\neg(\Box_B red(Bernard) \vee \Box_B\neg red(Bernard))$
resolvent	$\Box_{\mathrm{MB}}\Diamond_B\Diamond_A red(Charlotte)$
given	$\Box_{\mathrm{MB}}\Box_B(\Box_A red(Charlotte) \vee \Box_A\neg red(Charlotte))$
	by $\Box_{\mathrm{MB}}(\Box_A red(Charlotte) \vee \Box_A\neg red(Charlotte))$
resolvent	$\Box_{\mathrm{MB}}\Diamond_B\Box_A red(Charlotte)$
given	$\Box_{\mathrm{MB}}\Box_B(\Diamond_A\neg red(Charlotte) \vee red(Charlotte))$
	by $\Box_{\mathrm{MB}}(\Box_x rY \Rightarrow rY)$
resolvent	$\Box_{\mathrm{MB}}\Diamond_B red(Charlotte)$
given	$\Box_{\mathrm{MB}}(\Box_B red(Charlotte) \vee \Box_B\neg red(Charlotte))$
resolvent	$\Box_{\mathrm{MB}}\Box_B red(Charlotte)$

given	$\Box_{\mathrm{MB}}(\Diamond_B \neg red(Charlotte) \vee red(Charlotte))$
	by $\Box_{\mathrm{MB}}(\Box_x rY \Rightarrow rY)$
resolvent	$\Box_{\mathrm{MB}} red(Charlotte)$
implies	$\Box_C red(Charlotte)$

To be fair: the program loops if the full theory of mutual belief is included. The reason behind this is that the full mutual belief clauses sometimes act as a world generator, i.e. they are always able to generate new worlds such that any later goal which for some reason must fail and therefore backtracks will be called again and again thus producing all worlds which can possibly be generated. However, this is actually a problem of PROLOG itself. Solutions to such problems are therefore out of the scope of this paper.

Appendix A

Summary

In this chapter we summarize the PROLOG clauses which have to be added to the database for the most common modal logics and repeat the respective formula morphisms. The first part contains the theory axioms for constant domain modal logics and the second part provides the necessary E-clauses which capture the varying domain case. These are the extra clauses to be added to the clause set of the constant domain case. For readability we omitted to indicate the sorts of the respective variables since different sorts cannot be mixed up during the PROLOG derivations anyway as it was explained earlier.

A.1 Constant Domains

A.1.1 Serial Modal Logics

Let Φ be an arbitrary modal logic formula in negation normal form.
The formula morphism $\Pi^\star_{\mathrm{F}}$ for the constant domain, serial modal logic case is then defined as:

$$\begin{array}{lcl}
\Pi^\star_{\mathrm{F}}(x,u) & = & x \\
\Pi^\star_{\mathrm{F}}(f(t_1,\ldots,t_n),u) & = & f'(u,\Pi^\star_{\mathrm{F}}(t_1,u),\ldots,\Pi^\star_{\mathrm{F}}(t_n,u)) \\
\Pi^\star_{\mathrm{F}}(P(t_1,\ldots,t_n),u) & = & P'(u,\Pi^\star_{\mathrm{F}}(t_1,u),\ldots,\Pi^\star_{\mathrm{F}}(t_n,u)) \\
\Pi^\star_{\mathrm{F}}(\neg P(t_1,\ldots,t_n),u) & = & \neg P'(u,\Pi^\star_{\mathrm{F}}(t_1,u),\ldots,\Pi^\star_{\mathrm{F}}(t_n,u)) \\
\Pi^\star_{\mathrm{F}}(\Phi\vee\Psi,u) & = & \Pi^\star_{\mathrm{F}}(\Phi,u)\vee\Pi^\star_{\mathrm{F}}(\Psi,u) \\
\Pi^\star_{\mathrm{F}}(\Phi\wedge\Psi,u) & = & \Pi^\star_{\mathrm{F}}(\Phi,u)\wedge\Pi^\star_{\mathrm{F}}(\Psi,u) \\
\Pi^\star_{\mathrm{F}}(\forall x\ \Phi,u) & = & \forall x\ \Pi^\star_{\mathrm{F}}(\Phi,u) \\
\Pi^\star_{\mathrm{F}}(\exists x\ \Phi,u) & = & \exists x\ \Pi^\star_{\mathrm{F}}(\Phi,u) \\
\Pi^\star_{\mathrm{F}}(\Box\Phi,u) & = & \forall v{:}W\ R(u,v)\Rightarrow\Pi^\star_{\mathrm{F}}(\Phi,v)
\end{array}$$

Logic	Synonym	Extra-Clauses
KD	none	$R(u, ux)$.
KT	T, M	$R(u, u)$. $R(u, ux)$.
KD4	none	$R(u, ux)$. $R(u, vx) \leftarrow R(u, v)$.
KDB	none	$R(u, ux)$. $R(ux, u)$.
KDE	none	$R(\iota, x)$. $R(ux, vy)$.
KT4	S4	$R(u, u)$. $R(u, vx) \leftarrow R(u, v)$.
KTB	B	$R(u, u)$. $R(u, ux)$. $R(ux, u)$.
KTB4	S5, KT5, KDB4	$R(u, v)$.
KD4E	weak S5	$R(u, vx)$.

Table A.1: **Constant Domain, Serial Modal Logics**

$$\begin{aligned} \Pi^\star_{\mathrm{F}}(\Diamond\Phi, u) &= \exists x\colon F_R\, \Pi^\star_{\mathrm{F}}(\Phi, ux) \\ &\text{with initial call } \Pi^\star_{\mathrm{F}}(\Phi, \iota) \end{aligned}$$

The theory clauses to be added to the PROLOG database can be found in Table A.1. Note that, as shown in Section 5.1.1, adding the extra unit clause for KD is not necessary if the translation is changed accordingly. The same obviously holds for the modal logics KTB4 and KD4E for their background theory is also reflected by a single unit clause.

A.1.2 Non-Serial Modal Logics

For non-serial modal logics the formula morphism does not change considerably. The main difference lies in the extra literals which cover the possibility of having non-normal worlds.

Table A.2 contains the PROLOG clauses which are to be added to the result of the formula morphism.

Logic	Synonym	Extra-Clauses
K	none	$R(u, ux) \leftarrow N(u).$
K4	none	$R(u, ux) \leftarrow N(u).$ $R(u, vx) \leftarrow N(v), R(u, v).$
KB	none	$R(u, ux) \leftarrow N(u).$ $R(ux, u) \leftarrow N(u).$
KE	none	$R(\iota, x) \leftarrow N(\iota).$ $R(ux, vy) \leftarrow N(u), N(v).$
KB4	none	$R(u, v) \leftarrow N(\iota).$
K4E	K45	$R(u, vx) \leftarrow N(v).$
KBE	KB5, KB45, KB4E	$R(u, v) \leftarrow N(\iota)$

Table A.2: **Constant Domain, Non-Serial Modal Logics**

$$
\begin{array}{lcl}
\Pi^\star_{\mathrm{F}}(x, u) & = & x \\
\Pi^\star_{\mathrm{F}}(f(t_1, \ldots, t_n), u) & = & f'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(P(t_1, \ldots, t_n), u) & = & P'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(\neg P(t_1, \ldots, t_n), u) & = & \neg P'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(\Phi \vee \Psi, u) & = & \Pi^\star_{\mathrm{F}}(\Phi, u) \vee \Pi^\star_{\mathrm{F}}(\Psi, u) \\
\Pi^\star_{\mathrm{F}}(\Phi \wedge \Psi, u) & = & \Pi^\star_{\mathrm{F}}(\Phi, u) \wedge \Pi^\star_{\mathrm{F}}(\Psi, u) \\
\Pi^\star_{\mathrm{F}}(\forall x\ \Phi, u) & = & \forall x\ \Pi^\star_{\mathrm{F}}(\Phi, u) \\
\Pi^\star_{\mathrm{F}}(\exists x\ \Phi, u) & = & \exists x\ \Pi^\star_{\mathrm{F}}(\Phi, u) \\
\Pi^\star_{\mathrm{F}}(\Box\Phi, u) & = & \forall v{:}\, W\ R(u, v) \Rightarrow \Pi^\star_{\mathrm{F}}(\Phi, v) \\
\Pi^\star_{\mathrm{F}}(\Diamond\Phi, u) & = & N(u) \wedge \exists x{:}\, F_R\ \Pi^\star_{\mathrm{F}}(\Phi, ux)
\end{array}
$$

with initial call: $\Pi^\star_{\mathrm{F}}(\Phi, \iota)$

A.2 Varying Domains

A.2.1 Serial Modal Logics

$$
\begin{array}{lcl}
\Pi^\star_{\mathrm{F}}(x, u) & = & x \\
\Pi^\star_{\mathrm{F}}(f(t_1, \ldots, t_n), u) & = & f'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(P(t_1, \ldots, t_n), u) & = & P'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(\neg P(t_1, \ldots, t_n), u) & = & \neg P'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u))
\end{array}
$$

$$\begin{array}{lcl}
\Pi^\star_{\mathrm{F}}(\Phi \vee \Psi, u) & = & \Pi^\star_{\mathrm{F}}(\Phi, u) \vee \Pi^\star_{\mathrm{F}}(\Psi, u) \\
\Pi^\star_{\mathrm{F}}(\Phi \wedge \Psi, u) & = & \Pi^\star_{\mathrm{F}}(\Phi, u) \wedge \Pi^\star_{\mathrm{F}}(\Psi, u) \\
\Pi^\star_{\mathrm{F}}(\forall x\ \Phi, u) & = & \forall x\ E(u,x) \Rightarrow \Pi^\star_{\mathrm{F}}(\Phi, u) \\
\Pi^\star_{\mathrm{F}}(\exists x\ \Phi, u) & = & \exists y{:}\, F_E\ \Pi^\star_{\mathrm{F}}(\Phi, u)[x/uy] \\
\Pi^\star_{\mathrm{F}}(\Box\Phi, u) & = & \forall v{:}\, W\ R(u,v) \Rightarrow \Pi^\star_{\mathrm{F}}(\Phi, v) \\
\Pi^\star_{\mathrm{F}}(\Diamond\Phi, u) & = & \exists x{:}\, F_R\ \Pi^\star_{\mathrm{F}}(\Phi, ux) \\
\text{with initial call: } \Pi^\star_{\mathrm{F}}(\Phi, \iota) & &
\end{array}$$

The theory clause is the simple unit clause: $E(u, ux)$.

A.2.2 Non-Serial Modal Logics

$$\begin{array}{lcl}
\Pi^\star_{\mathrm{F}}(x, u) & = & x \\
\Pi^\star_{\mathrm{F}}(f(t_1, \ldots, t_n), u) & = & f'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(P(t_1, \ldots, t_n), u) & = & P'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(\neg P(t_1, \ldots, t_n), u) & = & \neg P'(u, \Pi^\star_{\mathrm{F}}(t_1, u), \ldots, \Pi^\star_{\mathrm{F}}(t_n, u)) \\
\Pi^\star_{\mathrm{F}}(\Phi \vee \Psi, u) & = & \Pi^\star_{\mathrm{F}}(\Phi, u) \vee \Pi^\star_{\mathrm{F}}(\Psi, u) \\
\Pi^\star_{\mathrm{F}}(\Phi \wedge \Psi, u) & = & \Pi^\star_{\mathrm{F}}(\Phi, u) \wedge \Pi^\star_{\mathrm{F}}(\Psi, u) \\
\Pi^\star_{\mathrm{F}}(\forall x\ \Phi, u) & = & \forall x\ E(u,x) \Rightarrow \Pi^\star_{\mathrm{F}}(\Phi, u) \\
\Pi^\star_{\mathrm{F}}(\exists x\ \Phi, u) & = & \exists y{:}\, F_E\ \Pi^\star_{\mathrm{F}}(\Phi, u)[x/uy] \\
\Pi^\star_{\mathrm{F}}(\Box\Phi, u) & = & \forall v{:}\, W\ R(u,v) \Rightarrow \Pi^\star_{\mathrm{F}}(\Phi, v) \\
\Pi^\star_{\mathrm{F}}(\Diamond\Phi, u) & = & N(u) \wedge \exists x{:}\, F_R\ \Pi^\star_{\mathrm{F}}(\Phi, ux) \\
\text{with initial call: } \Pi^\star_{\mathrm{F}}(\Phi, \iota) & &
\end{array}$$

With theory clause $E(u, ux)$ again.

A.2.3 Increasing and Decreasing Domains

The formula morphism does not change for these special cases, however there is a further theory clause which has to be contained in the clause set, namely:

$$E(ux, y) \leftarrow E(u, y) \text{ for the case of increasing domains}$$

and

$$E(u, y) \leftarrow E(ux, y) \text{ for the case of decreasing domains}$$

which just states that any domain element y which exists in world u (ux) also exists in world ux (u respectively).

Note that the theory axiom for decreasing domains may lead to infinite loops for input formulae which are not theorems. This does not hold, however, for K5 or stronger ones (as e.g. K45 or KD5). Here the theory clauses can be simplified to $E(ux, y)$ and $E(\iota, [x])$ in the serial, increasing domain case, and to $E(ux, y) \leftarrow N(u)$

and $E(\iota, [x])$ in the non-serial, increasing domain case. For decreasing domains we analogously get $E(u, vxy)$ and $E(\iota, [x])$ (serial) and $E(u, vxy) \leftarrow N(v)$ and $E(\iota, [x])$ (non-serial) which cannot run into endless loops.

References

[1] Martìn Abadi and Zohar Manna. Modal theorem proving. In *Proceedings 8th CADE, LNCS 230*. Springer, 1986.

[2] B. Chellas. *Modal Logic: An Introduction.* Cambridge University Press, 1980.

[3] Luis Farinas del Cerro. A simple deduction method for modal logic. *Information Processing Letters*, 14(2), 1982.

[4] Melvin Fitting. *Proof Methods for Modal and Intuitionistic Logics.* Reidel, 1983.

[5] G. Hughes and M. Cresswell. *An Introduction to Modal Logic.* Menthuen, London, 1968.

[6] R. Moore. *Reasoning About Knowledge and Action.* PhD thesis, MIT, Cambridge, 1980.

[7] Hans Jürgen Ohlbach. *A Resolution Calculus for Modal Logics.* PhD thesis, University of Kaiserslautern, Germany, 1989.

[8] Hans Jürgen Ohlbach. Semantics-based translation methods for modal logics. *Journal of Logic and Computation*, 1(5):691–746, 1991.

[9] Krister Segerberg. An essay in classical modal logic. Technical Report 13, University of Uppsala, Filosofiska Studier, 1971. Volume 1-3.