Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern

SEKI - REPORT

Context Logic

Hans Jürgen Ohlbach

SEKI Report SR-89-08

Context Logic

Hans Jürgen Ohlbach
Fachbereich Informatik, Universität Kaiserslautern
Postfach 3049, D-6750 Kaiserslautern, W.-Germany

This research was supported by the Deutsche Forschungsgemeinschaft, SFB 314 (D2)

Context Logic

Hans Jürgen Ohlbach
FB Informatik, University of Kaiserslautern
uucp: ...seismo!unido!uklirb!ohlbach

Abstract Context Logic (CL) is a logic in the original sense, but more than that, it is a methodology for designing a certain class of logics in such a way that automatically a first-order many-sorted resolution and paramodulation calculus is obtained. This calculus can be executed on a clause based predicate logic theorem prover. The class of logics which can be handled with the CL-methodology is mainly characterized by the existence of "hidden parameters", parameters like worlds in modal logics defining the context in which the terms and formulae are to be interpreted. The hidden parameters are usually determined implicitly by additional logical operators like for example $\Box$ (necessarily) and $\Diamond$ (possibly) in modal logic. These operators refer to an underlying semantical structure - Kripke's possible worlds structure in the case of modal logic, time points and time intervals in the case of temporal logic are examples. CL provides a means for axiomatizing these structures and for expressing the semantics of the desired operators in a formal language. This information about the desired logic is sufficient to translate formulae written in the operator syntax automatically into predicate logic syntax where the operators are replaced by quantifiers and the hidden parameters are made an explicit part of the formula. After the translation, information about a whole bunch of nested operators is shifted into one "context term" that can be handled by an appropriate unification algorithm. Hence, a resolution step may exploit information about many nested operators at once and is therefore much more goal directed than a corresponding step in a tableaux system for example.
The main limits of CL are:

- Since predicate logic is the "target logic" into which the designed logic is mapped, in order to obtain a complete calculus, its semantical structure must be first-order axiomatizable. This excludes certain properties like discreteness and finiteness.
- Due to the current limits of predicate logic resolution (no partial functions allowed for example) two further assumptions are still necessary. For Kripke structures these are the constant-domain assumption and the seriality assumption. For other structures the assumptions are analogous.

In order to demonstrate the method, a quite complex first-order many-sorted multi modal logic with operators indexed with arbitrary (possibly non-ground) terms is constructed using the CL tools. The logic is actually an extension of Clarke and Emerson's CTL temporal logic. Therefore, as a side effect, we get a proof theory for CTL.

Keywords: Automated Theorem Proving by Translation and Refutation, Resolution, Nonclassical Logics, Modal Logic, Temporal Logic, Process Logic, Epistemic Logic, Action Logic, CTL.

This work was supported by the Sonderforschungsbereich 314 of the German Science Foundation (DFG) and the ESPRIT Project 1033, FORMAST, of the European Community. The paper was written during a research stay at the Automated Reasoning Project of the Australian National University, Canberra.

Table of Contents

Chapter One

Introduction

1.1 Logics with "Context Semantics"

Á large class of nonclassical logics like modal, temporal, epistemic, action logics etc. have one important feature in common. This is the existence of "hidden parameters" which refer to an underlying semantical structure. The hidden parameters define the *context* in which terms and formulae are to be interpreted. They are implicitly determined by additional logical operators like for example $\Box$ (necessarily) and $\Diamond$ (possibly) in modal logic. A typical example for an underlying semantical structure is Kripke´s possible worlds structure as a semantical basis for modal logics. A formula $\Box P$ for example is interpreted: P holds in all worlds $\mathcal{W}_2$ which are accessible from the current world $\mathcal{W}_1$. $\Diamond P$ is interpreted: From the current world $\mathcal{W}_1$ there exists an accessible world $\mathcal{W}_2$ and P holds in $\mathcal{W}_2$. That means the interpretation of the predicate symbol P depends on the actual world. P may be true in a world $\mathcal{W}_1$ and false in a world $\mathcal{W}_2$. In other words P behaves like a one place predicate P(...) that depends on a "world parameter". Another example for a logic with hidden parameters and an underlying semantical structure is temporal logic with time points and time intervals. An expression "Tom is running" or more formally running(Tom) implicitly depends on the points in time. It may be true now and false in 5 minutes. "Tom runs 100m" or running-a-distance(Tom, 100m) respectively, however does not depend on a time point, but on a time interval. The statement "During Tom´s 100m runs he is always running" (which may be false when he occasionally stops for a while) may be encoded as "running-a-distance(Tom,100m) $\Rightarrow$ *during* running(Tom)". The formula shows that it may make sense to have simultaneously two types of contexts, namely time points and time intervals which are correlated by an operator *during* that quantifies over all time points in the current time interval.

Having the underlying semantical structure in mind, it is usually not hard to define syntax and a model theoretic semantics of a corresponding logic. To define an appropriate deduction calculus, however, is a much harder task. The problem is that the properties of the semantical structure, although it may be a simple thing like a linearly ordered set together with all its intervals, have to be characterized in a very indirect way by using axiom schemes in a syntax that does not talk at all about the semantical structure directly but uses some obscure operators. Who would for example guess that the transitivity of the accessibility relation in a Kripke structure has to be encoded into the modal logic axiom scheme $\Box\mathcal{F} \Rightarrow \Box\Box\mathcal{F}$?

When this is so complicated, why strive so hard with these operators? Why don´t we just use predicate logic, axiomatize the underlying semantical structure directly, add the corresponding parameters explicitly to the predicate and function symbols and replace the operators by corresponding quantifiers? There are four main reasons against this solution:

- Historical Reasons
  Modal logics for example have been developed on the basis of Hilbert calculi decades before Kripke discovered a model theoretic semantics.
- Pragmatic Reasons
  The operator syntax is usually much more intuitive and easier to use than the predicate logic syntax.
- Efficiency Reasons
  The operator syntax is usually less expressive than the corresponding predicate logic syntax. Therefore a calculus working on the operator syntax may be more efficient than a corresponding predicate logic version.
- Theoretical Reasons
  There are semantical structures which cannot be axiomatized in first-order logic. Discrete and finite Kripke structures are of this kind.

The method we are going to present in this work is aimed to overcome the pragmatic and efficiency arguments. It uses *theorem proving by translation and refutation* which allows to keep the operator syntax as a user friendly surface language, but internally it translates the formulae into predicate logic in such a way that a more efficient resolution [Robinson 65] and paramodulation [Robinson & Wos 69] calculus is applicable.

1.2 Theorem Proving by Translation and Refutation - An Analogy to Compilation of Programs.

Already in the early days of computer science it has turned out that much efficiency can be gained in the execution of programs written in highlevel programming languages by translating the programs into a lowlevel language where a more efficient interpreter (processor) is available. Originally programs were translated directly into the operation code of a processor. Nowadays programs are usually not translated directly into an operation code, but first into an intermediate language such as C and then with a standard compiler into the operation code. One of the reasons is that the development of compilers with less lowlevel languages as target language is in general easier and less susceptible to errors.

The analogy to a programming language - syntax and denotational semantics - is a logic - syntax and model theoretic semantics - and the analogy to an interpreter for a programming language is a deduction calculus for a logic. The idea which suggests itself is to look also for an analogy to a compiler. A compiler from a "source logic" SL into a "target logic" TL should translate an SL-formula $\mathcal{F}$ into a TL-formula $\mathcal{F}'$ such that $\mathcal{F}$ has an SL-model if and only if $\mathcal{F}'$ has a TL-model. If this condition holds, a refutation theorem prover for SL can be obtained simply by using an appropriate theorem prover for TL to refute the translated SL-formulae. One advantage is that a TL-theorem prover can be used for several source logics. Whereas highlevel programming languages as well as a highlevel logics are designed for the human user, a compiler is free to rearrange chunks of information to serve the underlying machinery, and not the human reader, in the optimal way. The efficiency gained through compilation of programs proves that operating on the original syntax is not always optimal, and this holds for logics as well. If therefore in addition the TL-calculus turns out to be even more efficient than an SL-calculus we have not only saved the effort to develop a specific SL-theorem prover, in fact our laziness is rewarded with a more efficient theorem prover.

Several "translation calculi" have been developed so far, mainly for classical modal logics as source logics and predicate logic as target logic with the aim to use predicate logic resolution theorem provers [Ohlbach 88], [Fariñas&Herzig 88], [Enjalbert&Auffray 88], [Moore 80] etc. In these systems a one-step translation from the source logic to the target logic has been developed, similar to the one-step translation from highlevel programming languages to the operation code. With this approach the translation algorithm and in particular the soundness and completeness proof has to be developed new for every new logic. A closer look at the different translation systems, however, has shown that a certain part of the translation is common to all source logics. Therefore it is advantageous to split the translation into two steps, the first one depending on the particular source logic and the second one common to all source logics. For this purpose an intermediate logic is necessary. This logic, Context Logic, will be presented in this paper. Here again we have the analogy to the compilation of programs. CL corresponds to the intermediate language, as for example C, which is used in two-step translations. We can summarize the analogies as follows:

Earlier approach:

Compilation of programming languages:

$$\begin{tikzcd} PL_1 \arrow[dr, "\text{compilation}"'] & \cdots & PL_n \arrow[dl] & \text{inefficient interpreter} \\ & \text{operation code} & & \text{efficient interpeter} \end{tikzcd}$$

Translation of logics:

$$\begin{tikzcd} SL_1 \arrow[dr, "\text{translation}"'] & \cdots & SL_m \arrow[dl] & \text{inefficient calculus} \\ & TL & & \text{efficient calculus} \end{tikzcd}$$

A more economic approach, proposed in this paper:

Compilation of programming languages:

$$\begin{tikzcd} PL_1 \arrow[dr] & \cdots & PL_n \arrow[dl] \\ & C \arrow[d] & \\ & \text{operation code} & \end{tikzcd}$$

Translation of logics:

$$\begin{tikzcd} SL_1 \arrow[dr, "\text{SL-depending translation}"'] & \cdots & SL_m \arrow[dl] \\ & CL \arrow[d, "\text{standard translation}"] & \\ & \text{predicate logic} & \end{tikzcd}$$

As compilation from C into different operation codes is possible, translation from CL into different logics is also possible. So far, however, only a translation into a particular order-sorted predicate logic as target logic has been developed.

1.3 One-Step Translations

In order to get an idea how translations for logics work, let us first have a look at algorithms for translating modal logic in one step into predicate logic with *resolution* as a basic calculus. There are different possibilities yielding translated formulae of different structure and search spaces of different size.

1.3.1 Relational Translation

There is a very easy way to translate modal logic formulae into predicate logic [Moore 80]: A special binary predicate symbol $\mathcal{R}$ is introduced which represents the accessibility relation. A formula $\Box\mathcal{F}$ is then translated into $\forall w\ \mathcal{R}(a,w) \Rightarrow \mathcal{F}[w]$ where 'a' denotes the current world and $\mathcal{F}[w]$ means adding 'w' as an additional argument to the terms and literals. Analogously $\Diamond\mathcal{F}$ is translated into $\exists w\ \mathcal{R}(a,w) \land \mathcal{F}[w]$. The properties of the accessibility relation can be expressed by simply adding the corresponding axioms for $\mathcal{R}$ to the formulae.

For example the formula $\Diamond\Diamond\forall x(\Diamond Px \land \Box Qx)$ is translated into the predicate logic formula

$$\exists a\ \mathcal{R}(0,a) \land \exists b\ \mathcal{R}(a,b) \land \forall x\ (\exists c\ \mathcal{R}(b,c) \land P(c,x) \land \forall w\ \mathcal{R}(b,w) \Rightarrow Q(w,x))$$

The problem with this *"relational"* method is that the actual world in which a term or literal is to be interpreted is not only determined by the term in the "world argument" of the predicates, for example the 'c' in P(c,x) above, but by the whole path of "world terms" leading to that particular term. This information, however, is spread over a whole bunch of $\mathcal{R}$-literals. One significant deduction step with a user defined predicate has therefore in general to be accompanied by several deduction steps which reason about worlds alone. The usual control strategies for resolution can not recognize these correspondences and may therefore easily get lost in irrelevant branches of the search space.

1.3.2 Functional Translation

In order to overcome this weakness, at least for some modal logics, a different translation technique has been developed where the relevant information about the actual world is concentrated in one single term [Ohlbach 88] [Fariñas&Herzig 88], [Enjalbert&Auffray 88]. In my system for example the above formula would be translated into $\exists a,b\ \forall x\ (\exists c\ P([abc],x) \wedge \forall u\ Q([abu],x))$
yielding $\forall x,u\ P([abc(x)],x) \wedge Q([abu],x))$ after Skolemization,
where the "context access terms" [abc(x)] and [abu] describe the complete path through the Kripke structure from the initial world to the actual world.

One of the main problems in the development of a semantics for these special terms was to handle the fact that the modal operators are some kind of dynamic operators. The set of objects over which they quantify depends on the current position in the Kripke structure. For example $\Box\mathcal{F}$ quantifies over all worlds accessible from the *current* world, and this world is determined by the embracing modal operators and quantifiers. The key idea for getting rid of this dynamic aspect was to translate modal operators into quantifiers over *functions* mapping worlds to accessible worlds. The set of such "world access functions", or more general "context access functions", is constant in each interpretation whereas the set of worlds they access from a given world may change from world to world. This allows to keep the operator's modal logic spirit, but to treat them technically as ordinary predicate logic quantifiers, quantifying over a fixed set of entities.

To realize this idea a two-sorted predicate logic with the two sorts D for domain elements and 'W→W' for context access functions is necessary. In the formula

$\forall x,u\ P([abc(x)],x) \wedge Q([abu],x))$

for example a, b are now constant symbols of sort 'W→W',
c is a function symbol of sort D → 'W→W',
x is a variable symbol of sort D and
u is a variable symbol of sort 'W→W'.

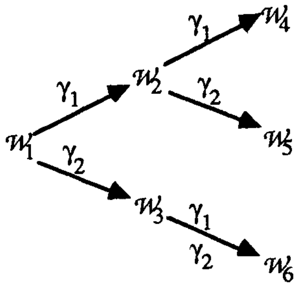
Strings of 'W→W'-terms are now interpreted as composition of context access functions. If for example a, b and u are interpreted as the context access functions γ_a, γ_b and γ_u then [abu] denotes the function $\gamma_a \circ \gamma_b \circ \gamma_u$ which maps the initial world in three steps to the actual world.

Correlations between the Accessibility Relation and the Context Access Functions

Since different modal logics are usually distinguished by the properties of the accessibility relation, and since we want to represent the accessibility relation $\Re$ by a set $\mathcal{CF}$ of context access functions, the proper correlations between $\Re$ and $\mathcal{CF}$ have to be established. The basic idea is to represent a binary relation $\Re$ as the argument-value relation of a set $\mathcal{CF}$ of one-place functions, i.e.

$$\forall w_1,w_2\ \Re(w_1,w_2) \text{ iff } \exists\gamma\in \mathcal{CF}\ w_2=\gamma(w_1)$$

Example:



relational representation:	functional representation:	
$\Re(w_1,w_2)$	$\{\gamma_1,\gamma_2\}$	
$\Re(w_1,w_3)$	γ_1: $w_1\to w_2$	γ_2: $w_1\to w_3$
$\Re(w_2,w_4)$	$w_2\to w_4$	$w_2\to w_5$
$\Re(w_2,w_5)$	$w_3\to w_6$	$w_3\to w_6$
$\Re(w_3,w_6)$		

Given a relation $\Re$, a corresponding set $\mathcal{CF}$ of one place functions - which is not necessarily unique - can be constructed such that the argument-value relation is just $\Re$, and, the other way round, given a set $\mathcal{CF}$ of one place functions on the set of worlds, their argument-value relation constitutes an accessibility relation. Since $\Re$ and $\mathcal{CF}$

are correlated, there must also be correlations between their properties. One correlation is obvious: If $\mathcal{CF}$ contains only total functions then $\mathfrak{R}$ is a *serial* relation where each world has an accessible world. On the other hand, if $\mathfrak{R}$ is serial then there is always a set of *total* context access functions. In the sequel we shall always assume that serial relations are represented with total context access functions. Another obvious correlation is: If $\mathfrak{R}$ is tree like then $\mathcal{CF}$ consists of injective functions only. (The other direction does not hold.) Further correlations are:

- reflexivity of $\mathfrak{R}$ $\leftrightarrow$ there is always a set $\mathcal{CF}$ containing the identity function.
- transitivity of $\mathfrak{R}$ $\leftrightarrow$ there is always a set $\mathcal{CF}$ which is closed under composition.
- symmetry of $\mathfrak{R}$ $\leftrightarrow$ there is always a set $\mathcal{CF}$ containing for each function its inverse.

To get a complete resolution calculus for translated modal formulae, these properties have to be exploited. A first possibility to do this is to axiomatize the sort 'W→W' explicitly. For example the reflexivity requires the axiom
$\exists id$:'W→W' $\forall x$:'W→W' $id \circ x = x \circ id = x$.
(We us an explicit composition function symbol $\circ$ instead of the syntax with brackets.)
The transitivity of $\mathfrak{R}$ is expressed by the associativity of $\circ$ and the sort declaration $\circ$:'W→W' × 'W→W' → 'W→W' expressing that $\mathcal{CF}$ is closed under functional composition, or, with other words, that each world which is accessible in n steps is also accessible in one step. (In the nontransitive case this declaration has to be $\circ$:'W→W' × 'W→W' → $\perp$ where $\perp$ denotes some error sort. This declaration prevents 'W→W'-variables to be instantiated with terms $s \circ t$ which denote functions accessing worlds in two or more steps.) The symmetry of $\mathfrak{R}$ is axiomatized by introducing an inverse function $^{-1}$:W→W' → 'W→W' with the corresponding axiom.

The disadvantage of the explicit axiomatization is that equations occur and equations are difficult to handle in a normal resolution theorem prover. Fortunately for the above cases the equations can be completely replaced by corresponding theory unification algorithms such that equality handling is no longer necessary. Algorithms are for example given in [Ohlbach 88].

The theory unification algorithms can handle the context access terms efficiently because the relevant part of the Kripke structure is at their disposal. One resolution step in the resolution calculus may invoke information about several nested modal operators and quantifiers in the original formula at once and therefore correspond to a number of deduction steps in a tableaux or sequent calculus. This allows for much bigger steps in the proof search, thus reducing the search space considerably. Moreover, since worlds are represented as terms and unification is applied to these terms, instead of generating worlds explicitly one by one, as in some classical calculi, we stay always on the "most general world", which further shrinks the search space.

1.4 Two-Step Translations

The translation of modal formulae into predicate logic consists of several steps. First of all the operators have to be replaced by quantifications over context access functions. For example

$\Box\forall x{:}D\, \Diamond P(x, a)$ yields $\forall u$:'W→W' $\forall x{:}D$ $\exists v$:'W→W' $P(x, a)$

Second, the sequences of nested quantifications over context access variables have to be collected into context terms and attached as additional arguments to the terms and literals:

$\forall u$:'W→W' $\forall x{:}D$ $\exists v$:'W→W' $P(x, a)$ yields

$\forall u$:'W→W' $\forall x{:}D$ $\exists v$:'W→W' $P([u\ v], x, a([u\ v]))$.

Finally existentially quantified variables have to be Skolemized. Hence, an optimized Skolemization is possible which allows to make the Skolem functions for the context access variables independent of the the universally quantified context access variables. Thus, instead of

$\forall u$:'W→W' $\forall x{:}D$ $P([u\ f_v(u, x)], x, a([u\ f_v(u, x)]))$ we obtain

$\forall u$:'W→W' $\forall x{:}D$ $P([u\ f_v(x)], x, a([u\ f_v(x)]))$.

The last two steps do not depend on the particular kind of modal logic. They formalize the concept of "contexts" and "context access functions". Therefore it is a good idea to separate these three steps into the first step which depends on the particular source logic, and the last two steps which are independent of the source logic. For this purpose an intermediate logic, Context Logic, is necessary where formulae like $\forall u$:'W→W' $\forall x$:D $\exists v$:'W→W' P(x, a) make sense although the context access variables need not yet occur in the literals. Thus, CL is essentially a logic with predicate logic syntax but modal logic semantics. The translation from the source logic to CL consists mainly of expressing the semantics of the operators with CL-quantifications. The translation from CL into predicate logic, on the other hand, moves information from the quantifier level to the term level. The corresponding soundness and completeness proofs and the justification for the optimized Skolemization are therefore technically quite complex. They, however, can be settled once and forever.

1.5 Indexed Operators

Context Logic supports indexed operators as they are used in epistemic and action logics. The indices may be arbitrary - possibly non-ground - terms. Interpreting these operators as "belief operators" for example, it is then easy to formalize a statement like "everybody believes that his mother believes that her child is the best of the world" by

$$\forall x{:}\text{human}\ \Box_x\ \Box_{\text{mother}(x)}\ \text{best-of-the-world}(x).$$

The translation of this formula into CL yields

$$\forall x{:}\text{human}\ \forall{\downarrow}(u{:}\text{'D,W→W'}, x)\ \forall{\downarrow}(v{:}\text{'D,W→W'}, \text{mother}(x))\ \text{best-of-the-world}(x)$$

and the final translation into predicate logic yields

$$\forall x{:}\text{human}\ \forall u,v{:}\text{'D,W→W'}\ \text{best-of-the-world}({\downarrow}(u, x) \circ {\downarrow}(v, \text{mother}(x)), x).$$

u and v denote functions that map words to worlds, however depending on domain elements. $\downarrow$ is the application function symbol. Its type is $\downarrow$:'D,W→W' × D → 'W→W'. A term $\downarrow$(u, s) is therefore interpreted as a usual context access function which, however, describes transitions parametrized with the interpretation of s, a domain element.

1.6 Order-Sorted Logic as Target Logic

The translation calculi for classical modal logics in fact do not need a sorted logic as target logic. The two sorts D and 'W→W' mentioned above restrict the instantiation of variables. Once these constraints for the variable instantiation are built into the unification algorithms, the sorts can be ignored completely. For more complex source logics, however, the sort mechanism of the target logic becomes essential. To illustrate this, let us try to define a translation calculus for a multi modal logic, let us call it MML, with a pair $\Box^{\emptyset}$, $\Diamond^{\emptyset}$ of operators referring to a basic accessibility relation $\Re^{\emptyset}$, a pair $\Box^{r}$, $\Diamond^{r}$ of operators referring to the reflexive closure $\Re^{r}$ of $\Re^{\emptyset}$, furthermore operators $\Box^{r}$, $\Diamond^{r}$, $\Box^{s}$, $\Diamond^{s}$, $\Box^{t}$, $\Diamond^{t}$, $\Box^{rs}$, $\Diamond^{rs}$, $\Box^{rt}$, $\Diamond^{rt}$, $\Box^{st}$, $\Diamond^{st}$, $\Box^{rst}$ and $\Diamond^{rst}$ referring to the symmetric (s), transitive (t), reflexive-transitive (rt) etc. closures of $\Re^{\emptyset}$. With a temporal interpretation of the accessibility relation we can for example formalize a statement "Either I have the idea immediately or I'll never get it." in MML with $\Diamond^{r}$have(idea, I) $\vee$ $\Box^{rt}\neg$have(idea, I) where $\Diamond^{r}$ is interpreted as "possibly now or in the immediate future" and $\Box^{rt}$ is interpreted as "henceforth". The translated version is:

$$\exists x{:}\text{'W}\rightarrow^{r}\text{W'}\ \text{have}(x, \text{idea}, \text{I}) \vee \forall y{:}\text{'W}\rightarrow^{rt}\text{W'}\ \neg\text{have}(y, \text{idea}, \text{I}).$$

("idea" and "I" are *rigid* symbols. They do not depend on the worlds.)

In the functional translation for MML, a single set of context access functions is no longer sufficient. We need 'W$\rightarrow^{\emptyset}$W'-functions mapping worlds to $\Re^{\emptyset}$-accessible worlds, 'W$\rightarrow^{r}$W'-functions mapping worlds to $\Re^{r}$-accessible worlds etc. Furthermore we have to express that each 'W$\rightarrow^{\emptyset}$W'-function is also a 'W$\rightarrow^{r}$W'-function, a 'W$\rightarrow^{s}$W'-function, a 'W$\rightarrow^{t}$W'-function etc. These sets of functions can very easily be axiomatized in an order-sorted logic. The sort symbols 'W$\rightarrow^{\emptyset}$W', 'W$\rightarrow^{r}$W', 'W$\rightarrow^{s}$W', 'W$\rightarrow^{t}$W', 'W$\rightarrow^{rs}$W',

'$W \rightarrow^{rt} W$' and '$W \rightarrow^{st} W$', '$W \rightarrow^{rst} W$', are introduced and the sort hierarchy, a Boolean algebra with 8 elements expresses the subset relationships:

$$\begin{tikzcd}
 & \text{'}W\rightarrow^{rts}W\text{'} \arrow[dl, dash] \arrow[d, dash] \arrow[dr, dash] & \\
\text{'}W\rightarrow^{rt}W\text{'} \arrow[d, dash] \arrow[dr, dash] & \text{'}W\rightarrow^{rs}W\text{'} \arrow[dl, dash] \arrow[dr, dash] & \text{'}W\rightarrow^{ts}W\text{'} \arrow[dl, dash] \arrow[d, dash] \\
\text{'}W\rightarrow^{r}W\text{'} & \text{'}W\rightarrow^{t}W\text{'} & \text{'}W\rightarrow^{s}W\text{'} \\
 & \text{'}W\rightarrow^{\emptyset}W\text{'} \arrow[ul, dash] \arrow[u, dash] \arrow[ur, dash] &
\end{tikzcd}$$

The type declarations for the composition function symbol $\circ$ can be used to encode more information about the accessibility relation. For example the declaration $\circ$:'$W\rightarrow^{\emptyset}W$' $\times$ '$W\rightarrow^{\emptyset}W$' $\rightarrow$ '$W\rightarrow^{t}W$' expresses the fact that two single steps correspond to one step in the transitive closure. $\circ$:'$W\rightarrow^{\emptyset}W$' $\times$ '$W\rightarrow^{s}W$' $\rightarrow$ '$W\rightarrow^{rt}W$' expresses that one step followed by either one step forward or one step backward results in one step in the reflexive transitive closure. With declarations of this kind we can ensure that for example a variable of type '$W\rightarrow^{\emptyset}W$' can never be instantiated with a term $\circ(s, t)$ which is at least of type '$W\rightarrow^{t}W$'. Thus, order-sorted predicate logic with this kind of sorts gives us the possibility to handle for example modal logics where different modal operators corresponding to different closures of the accessibility relation are used *simultaneously.*

1.7 Axiomatization of Context Access Functions

The hierarchy of context access function sorts and the type declarations for the composition function are not yet sufficient to describe the context structure completely. Explicit axioms stating more than subset relationships are in general necessary. For example in order to express that the '$W\rightarrow^{r}W$'-functions really describe a reflexive relation we have to add an identity element which maps a world to itself. Thus, we need an axiom:

$$\exists id{:}\text{'}W\rightarrow^{r}W\text{'}\ \forall x{:}\text{'}W\rightarrow^{r}W\text{'}\ \ id \circ x = x \circ id = x.$$

Furthermore we want $\Re^{r}$ to be exactly the reflexive closure of $\Re^{\emptyset}$ and not more. Therefore an $\Re^{r}$-transition is either a $\Re^{\emptyset}$-transition or an identity transition. The "functional" axiom that expresses exactly this correlation is:

$$\forall x{:}\text{'}W\rightarrow^{r}W\text{'}\ \exists y{:}\text{'}W\rightarrow^{\emptyset}W\text{'}\ \forall w{:}W\ x(w) = y(w) \vee x(w) = w$$

(The sort W denotes the set of worlds.)
More axioms of this are needed for describing the other functional sorts in MML above.

Hence, a complete functional description of the source logic's semantical structure consists of

- a hierarchy of sorts describing the context access functions,
- the type declarations for certain symbols like $\circ$ and
- an axiomatization of the context access functions.

Let us now summarize the basic ideas behind Context Logic. CL is a means for designing new logics, let us call them SL, as extensions of first-order many-sorted predicate logic (with built-in equality reasoning) where the interpretation of terms and literals depends on some context. The context is an element or a tuple out of one ore more algebraic structures, the "context structures". Starting from an initial context, operators in the syntax of SL are used to jump from context to context until the "actual context" that is to be used for the interpretation of a subformula inside a nested formula is reached. For the calculus, the operator syntax, however, is only used as a user friendly surface syntax. Formulae in that syntax are translated in two steps into a pure predicate logic syntax, such that for example existing resolution and paramodulation calculi can be used. The first translation step, which actually translates into Context Logic, replaces operators by quantifiers over "context access" functions. The replacement rules for this step are defined just by writing down the semantics of the operator in the syntax of Context Logic. As an example, the replacement rules for the modal operators are:

$$\Psi(\Box\mathcal{F}) = \forall x{:}\text{'}W\rightarrow W\text{'}\ \Psi(\mathcal{F}) \qquad \text{and} \qquad \Psi(\Diamond\mathcal{F}) = \exists y{:}\text{'}W\rightarrow W\text{'}\ \Psi(\mathcal{F})$$

These rules have to be given by the designer of the logic. In the second translation step, which is done automatically by the Context Logic mechanism, the so quantified variables are collected to build "context terms" which are attached as additional "context parameters" at the terms and literals in order to get pure predicate logic formulae. The context structures themselves, i.e. the model theoretic semantics of SL, are to be axiomatized in Context Logic. This also has to be done once by the designer of the logic.

Context Logic has two main advantages:

- Using Context Logic, the design of a first-order logic including proof theory and all soundness and completeness proofs is not more work than the axiomatization of, say, boolean algebras in predicate logic.
- The Context Logic methodology fits into the paradigm of the predicate logic resolution and paramodulation principles. Therefore it is no longer necessary to write specialized theorem provers for the kind of nonclassical logics that can be handled by CL. That means most of the sophisticated representation and search control techniques - and even existing implementations - that have been developed for predicate logic theorem proving, and even logic programming, can immediately be applied. This is an indirect advantage which, however, should not be underestimated because developing a calculus is usually a matter of man weeks or at most man months, developing a powerful theorem prover for quantified logics, however, is a matter of man decades.

A Short Summary of the Subsequent Chapters

Chapter 2: Logic Morphisms

The main idea of Context Logic is to translate the theorems to be proved into a logic with a more efficient calculus. To provide adequate notions and notation for the translation operations we formally specify logics and mappings - morphisms - between logics. We establish a notion of *theorem proving by translation and refutation.*

Chapter 3: Order-Sorted Predicate Logic

Order-sorted predicate logic is the "target logic" into which the theorems to be proved are translated. Furthermore it provides the syntax for axiomatizing the context structures. We introduce the basic notions of order-sorted predicate logic as far as they are necessary for the CL formalism.

Chapter 4: Context Logic

Context Logic is an intermediate logic lying between the "source logic" and the "target logic", order sorted predicate logic. The translation from the "source logic" into Context Logic is the part which is specific for each source logic and has to be done by the designer of the source logic. The translation from Context Logic into predicate logic is the part which is common for all source logics and is described in this chapter. The role of Context Logic can be compared to the role, for example assembler languages or language like C play for the compilation of highlevel programming languages.

Chapter 5: Multi Modal Logic

To illustrate the Context Logic methodology and to demonstrate its usefulness, we design a specific "source logic" using the CL tools. It is an extension of the first-order modal logics D, T, D4 and S4, allowing simultaneous use of accessibility relations with different properties, indexed modal operators, a true 'eventually' operator and some 'until' operators. We give interpretations of multi modal logic as a temporal logic, a process logic, an action logic and an epistemic logic.

Chapter 6: Summary

The final conclusion of this work is that a large class of nonclassical logics can be handled efficiently with classical methods. We summarize the limits of the CL method and discuss further extensions.

Although most of the logical notions are formally defined within this work, we assume some familiarity with the standard predicate and modal logic as well as some knowledge about universal algebra and automated theorem proving. Some knowledge about the basic ideas of epistemic logics would also be helpful. Standard references are [Chang&Lee 73], [Fitting 83], [Grätzer 79], [Loveland 78], [Hintikka 62], [Hughes&Cresswell 68], [Smullyan 68].

Chapter Two

Logic Morphisms

The main idea of the Context Logic methodology is to realize a theorem prover for a given logic by translating the formula to be proved into a logic with an efficient calculus. This is essentially the same idea as the compilation idea frequently used in the design of programming languages. Highlevel programming languages are usually not interpreted directly, but compiled into an efficiently executable operation code of a processor. Unlike compiler building, the construction of translators for logics has not yet been systematized and supported by standard notions and methods. In this chapter we therefore try to systematize some of the well known notions about logics with respect to the description of translators for logics. The schemes we are going to develop should cover all kinds of two-valued logics with model theoretic semantics.

The kind of logics we are considering can be described by giving the syntax and its model theoretic semantics. The syntax is specified by describing the signature, i.e. the basic alphabet of nonlogical symbols, and by giving formation rules for terms and formulae. The description of the signature may already contain logical statements as for example the subsort declaration 'integer' $\sqsubseteq$ 'real' in a sorted logic. The formation rules for terms and formulae are in general also not so straightforward as in pure predicate logic. In some of the order-sorted logics very complex mechanisms have to ensure that the terms and formulae are well-sorted. The model theoretic semantics is usually defined in three steps. The first step is to define the signature interpretation, i.e. the interpretation of the nonlogical symbols. The signature interpretation itself is very often separated into the interpretation of the nonvariable symbols, which is the basic information necessary to interpret closed formulae, some context information as for example the initial world in modal logics, and into variable assignments which change dynamically when a quantified formula is interpreted. The second step is to turn the signature interpretation into an interpreter for terms by following the formation rules for terms. The last step is to define the satisfiability relation. The satisfiability relation actually fixes the meaning of the logical symbols and allows to evaluate formulae to 'true' or 'false'.

Definition 2.1 Logics

A (two-valued) **logic** (with model theoretic semantics) is a tuple (syntax, semantics) where **syntax** is a triple $(\mathbf{\Sigma}, \theta, \varphi)$ consisting of

- ➤ a set $\mathbf{\Sigma}$ of **signatures**,
- ➤ a function θ that maps a signature Σ to a set of **Σ-terms** (or terms for short) and
- ➤ a function φ that maps a signature Σ to a set of **Σ-formulae** (or formulae for short)

and **semantics** is a triple $(I, \Theta, \models)$ consisting of

- ➤ a function I that maps a signature Σ to the set of **signature interpretations** over Σ (or Σ-interpretations for short) - each signature interpretation consists of a **frame** $\mathfrak{F}$, a **context** $\mathcal{C}$ and a **variable assignment** $\mathcal{V}$ -,
- ➤ a function Θ that turns a signature interpretation into an **interpreter** for terms and
- ➤ a **satisfiability relation** $\models \in$ signature interpretations $\times$ formulae. ■

Example: With the above notions, pure predicate logic would be described as follows:

- A signature is a set of variable, function and predicate symbols. They are separated according to their arity. Σ is the set of all these signatures.
- The function θ is essentially the inductive definition of terms.
- The function φ is essentially the inductive definition of formulae.
- The function I assigns to each signature Σ the set of Σ-structures (which are essentially Σ-algebras) and variable assignments. Contexts are irrelevant for predicate logic.
- The function Θ turns a signature interpretation into an intepreter for terms by lifting variable assignments to the induced homomorphisms from the term algebra into the Σ-structure.
- $\models$ is the usual satisfiability relation. ■

A **specification** $(\Sigma, \mathcal{F})$ in a logic L is a signature Σ together with a set of Σ-formulae $\mathcal{F}$.

Definition 2.2 Satisfiability

- Given a logic L and an L-signature Σ, a Σ-formula $\mathcal{F}$ is called **L-satisfiable** (or simply satisfiable) iff $\Im \models \mathcal{F}$ for *some* signature interpretation $\Im$ ($\Im$ **satisfies** $\mathcal{F}$).
- A Σ-interpretation satisfies a specification $\mathcal{S} = (\Sigma, \mathcal{F})$ iff it satisfies all formulae in $\mathcal{F}$.
- A signature interpretation satisfying a formula or specification $\mathcal{S}$ is called a **model** for $\mathcal{S}$.
- $\mathcal{S}$ is **unsatisfiable** iff it is not satisfiable.
- A Σ-formula $\mathcal{F}$ is a **theorem** (or tautology) iff $\Im \models \mathcal{F}$ for *all* Σ-interpretations $\Im$. ■

Usually there is a notion of **closed formulae** in a logic. In a closed formula all variables are bound by quantifiers. Models for closed formulae are independent of variable assignments. That means whenever a closed formula $\mathcal{F}$ is satisfied by an interpretation $\Im = (\mathfrak{F}, \mathcal{C}, \mathcal{V})$ then $(\mathfrak{F}, \mathcal{C}, \mathcal{V}_1)$ satisfies $\mathcal{F}$ for all variable assignments $\mathcal{V}_1$. This is in general not the case for contexts. In modal logic, for example, satisfiability of closed formulae is usually defined relative to an initial context, i.e. an initial world. Therefore contexts are in general an essential part of models for formulae and specifications. Variable assignments are used in the satisfiability relation for recording (semantical) bindings to variables during a recursive descent into formulae.

We are now going to define logic morphisms as satisfiability preserving mappings between logics. They consist of a syntactial component, a mapping of signatures and formulae, and a semantical component, a mapping of interpretations. The syntactical component is essentially the "compiler" that translates specifications from one logic into another. The existence of the semantical component ensures that the syntactical translations map satisfiable specifications to satisfiable specifications (soundness) and unsatisfiable specifications to unsatisfiable specifications (completeness). With predicate logic as a target logic, a logic morphism allows **theorem proving** by **translation** (into predicate logic) **and refutation** (for example with predicate logic resolution and paramodulation).

Definition 2.3 Logic Morphisms

A **logic morphism** is a mapping Ψ between two logics $L_i = ((\Sigma_i, \theta_i, \varphi_i), (\mathcal{I}_i, \Theta_i, \models_i))$, i = 1,2.
It consists of the two components $(\Psi_{\mathcal{S}}, \Psi_{\mathfrak{I}})$ where

- $\Psi_{\mathcal{S}}$ is a **specification morphism** mapping L_1-specifications to L_2-specifications.
  $\Psi_{\mathcal{S}}$ may contain the two components,
  - Ψ_Σ, a signature morphism mapping L_1-signatures to L_2-signatures, and
  - $\Psi_{\mathcal{F}}$, a formula morphism mapping L_1-formulae to L_2-formula such that
    Σ_1-formulae are mapped to $\Psi_\Sigma(\Sigma_1)$-formulae, i.e. $\forall \Sigma \in \Sigma_1 : \mathcal{F} \in \varphi_1(\Sigma) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}) \in \varphi_2(\Psi_\Sigma(\Sigma))$

  (In general, $\Psi_{\mathcal{S}}$ not only translates formulae, but adds new symbols and formulae.)
- $\Psi_{\mathfrak{I}}$ is a bidirectional **interpretation morphism**, a mapping between L_1-interpretations and L_2-interpretations ensuring satisfiability preservation, i.e.
  - if an L_1-specification $\mathcal{S}$ is satisfied by $\mathfrak{I}_1$ then $\Psi_{\mathcal{S}}(\mathcal{S})$ is satisfied by $\Psi_{\mathfrak{I}}(\mathfrak{I}_1)$ (soundness) and
  - if $\Psi_{\mathcal{S}}(\mathcal{S})$ is satisfied by $\mathfrak{I}_2$ then $\mathcal{S}$ is satisfied by $\Psi_{\mathfrak{I}}^{-1}(\mathfrak{I}_2)$ (completeness) ■

Examples: Transformation into negation normal form and Skolemization is a logic morphism from predicate logic into the fragment of predicate logic without existential quantifier. Notice that only the preservation of satisfiability is required. Skolemization is the typical example that transforms tautologies not necessarily into tautologies. For example the tautology $\exists x Px \vee \forall x \neg Px$ is transformed into $Pa \vee \forall x \neg Px$ which is satisfiable, but not a tautology. Transformations of skolemized formulae into clauses is an example for a logic morphism which preserves tautologies. ■

Proposition 2.4 The composition of two logic morphisms is again a logic morphism.
The proof is straightforward. ■

This property allows to link translation steps together or, the other way round, to break complicated translations down into a sequence of simpler ones. We shall exploit this to decompose the translation from multi modal logic (MM-Logic) to order-sorted predicate logic (OSPL) into a first translation Ψ from MM-Logic to context logic (CL) and further translation Π from context logic to predicate logic.

$$\begin{tikzcd}
\text{MM} \arrow[drr, "\Psi"] & & & & \\
\text{other logics} \arrow[rr] \arrow[rr, bend right] & & \text{CL} \arrow[rr, "\Pi"] & & \text{OSPL}
\end{tikzcd}$$

mainly semantics mainly syntax

Π deals with the purely syntactical stuff of moving context information from the quantifier level or operator level respectively to the term level, whereas Ψ's work mainly consists of axiomatizing the context structure and is therefore closely oriented on the semantics of the source logic.

Chapter Three

Order-Sorted Predicate Logic

A number of many-sorted predicate logics have been developed so far. The different versions can be distinguished by the structure of the sort information they can represent, whether it is just a flat list of sorts, a semilattice or lattice, a feature sort structure etc., and by the kind of sort information for function and predicate symbols they can handle. The minimal requisites we need to make context logic running are: A hierarchical sort structure, i.e. a partial order, and overloaded (polymorphic) function sort declarations. As we have seen in the introduction, overloaded function sort declarations like $\circ$:‘$W\rightarrow^{\emptyset}W$’$\times$‘$W\rightarrow^{\emptyset}W$’$\rightarrow$‘$W\rightarrow^{t}W$’, $\circ$:‘$W\rightarrow^{\emptyset}W$’$\times$‘$W\rightarrow^{s}W$’$\rightarrow$‘$W\rightarrow^{rt}W$’ etc. are necessary for the composition function $\circ$. The currently most advanced many-sorted predicate logic with a full developed resolution and paramodulation calculus is that of Manfred Schmidt-Schauss [Schmidt-Schauss 88], an extension of Walther´s many-sorted predicate logic [Walther 87]. Schmidt-Schauss´ logic fulfills the requirements for Context Logic and even a bit more. In addition to overloaded function sort declarations of the above kind it allows so called “term declarations” where it is possible to declare for example that the sort of terms x:real*y:real are usually of sort ‘real’, but the special terms $x*x^{-1}$ and $x^{-1}*x$ are of sort ‘integer’. This special feature is not necessary for the CL mechanisms themselves, but it makes the logics which are built with CL more powerful.

Since Schmidt-Schauss´ logic is new and quite complex, we briefly introduce its main notions. We follow in principle the usual Tarski scheme for defining syntax and semantics of predicate logic, but we have to include extra devices for handling the sort information.

In the sequel $\mathcal{D}(f)$ denotes the the **domain** of the function f.

3.1 Syntax

The syntax of the order-sorted predicate logic (OSPL) consists of the basic signature elements as there are variable, function, predicate and sort symbols, furthermore there are declarations about the sort structure, formation rules for general terms and formulae as well as formation rules for the special class of *well sorted* terms and formulae.

An **unsorted signature** $\bar{\Sigma}$ consists of the three pairwise disjoint sets of symbols.

- ➤ F_Σ the set of **function symbols.**
- ➤ V_Σ the set of **variable symbols.**
- ➤ P_Σ the set of **predicate symbols.**

A **term declaration** is a pair (t,S) usually written as t:S, where t is a nonvariable term and S is a sort symbol. If t is of the form $f(x_1,\ldots,x_n)$, where the x_i are different variables, then we say t:S is a **sort declaration** for f, otherwise it is a **proper term declaration.** We sometimes abbreviate sort declarations $f(x_1,\ldots,x_n)$:S as $f{:}S_1\times\ldots\times S_n\rightarrow S$, where S_i is the sort of the variable x_i.
A **subsort declaration** has the form $R\sqsubseteq S$, where R and S are sorts. A **predicate declaration** is of the form $P{:}S_1\times\ldots\times S_n$, where the S_i are sorts.

Definition 3.1.1 Sorted Signatures

A (sorted) **signature** Σ consists of

- an unsorted signature $\bar{\Sigma}$,
- a set S_Σ of sorts
- a function S: $V_\Sigma \rightarrow S$ such that for every sort $S \in S_\Sigma$, there exist countably infinitely many variables $x \in V_\Sigma$ with $S(x) = S$,
- a set of subsort declarations, term declarations and predicate declarations.

We assume that the equality predicate $=_\Sigma$ is in P_Σ and that for all sorts R,S the predicate declaration $=_\Sigma$:R×S is also in Σ. ■

For a signature Σ let $\sqsubseteq_\Sigma$ be the quasi-ordering on S_Σ defined by the reflexive and transitive closure of the subsort declarations.

Definition 3.1.2 Well Sorted Terms, Atoms and Formulae

The set of **well sorted terms** $T_{\Sigma,S}$ of sort S in the signature Σ is (recursively) constructed by the following three rules:

- $x \in T_{\Sigma,S}$ if $S(x) \sqsubseteq_\Sigma S$
- $t \in T_{\Sigma,S}$ if $t{:}R \in \Sigma$ and $R \sqsubseteq_\Sigma S$
- $t[x/r] \in T_{\Sigma,S}$ if $t \in T_{\Sigma,S}$, $r \in T_{\Sigma,R}$ and $x \in V_\Sigma$ such that $R \sqsubseteq_\Sigma S(x)$.
  (t[x/r] means substituting r for x in t.)

The set T_Σ of all Σ-terms (well sorted terms) is defined as the union $\cup\{T_{\Sigma,S} \mid S \in S_\Sigma\}$.
The sort S(t) of a term t is the sort S of the greatest (with respect to set inclusion) $T_{\Sigma,S}$ containing t.
We sometimes use t:S to denote that S = S(t) is the sort of t.
An **atom** $P(t_1,\ldots,t_n)$ is **well sorted** if $t_i \in T_{\Sigma,Si}$ for $i=1,\ldots,n$ and $P{:}S_1\times\ldots\times S_n$ is a predicate declaration in Σ.

A **well formed formula** (well sorted) is a formula built with well sorted atoms and the logical connectives and quantifiers $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow, \forall, \exists$ in the usual way. We write $\forall x{:}S\ \mathcal{F}$ and $\exists x{:}S\ \mathcal{F}$ to indicate that the sort of the variable is S. A formula is called **closed** when all variables x occur in the scope of a quantifier $\forall x\ldots$ or $\exists x\ldots$. ■

For any object o, Vars(o) denotes the set of variables occurring in o.
We use a predicate $\in$ where $s \in t$ holds if s occurs as subterm in t.

Assumption

In the sequel we assume OSPL-specifications to contain the reflexive axiom for the equality symbol. This is necessary for the completeness of the paramodulation calculus in OSPL.

3.2 Algebras and Homomorphisms

Algebras and homomorphisms are the basic building blocks for the definition of the semantics of well sorted terms and well formed formulae. A Σ-algebra $\mathcal{A}$ for a signature Σ consists of a carrier set and a set of functions which correspond to Σ in the right way. The carrier set is divided into subsets according to Σ's sort structure and the domain-range relations of the functions in $\mathcal{A}$ match the corresponding term declarations. A special Σ-algebra is the algebra of free terms where the carrier set consists of the well sorted terms themselves and the functions are constructor functions for terms, i.e. they take n terms $t_1,\ldots,t_n$ and create a new term $f(t_1,\ldots,t_n)$. This fact can be exploited to define the semantics of terms just by an homomorphism from the free term algebra into a

corresponding Σ-algebra. Such an homomorphism is actually an interpreter which evaluates terms in a given algebra.
As an auxiliary definition we first introduce Σ-quasi-algebras as algebras which need not respect the subsort and term declarations in Σ.

Definition 3.2.1 Σ-Quasi-Algebras
A **Σ-quasi-algebra** $\mathcal{A}$ for a signature Σ consists of a carrier set A, a partial function $f_{\mathcal{A}}:A^{\text{arity}(f)} \to A$ (with domain $\mathcal{D}(f_A)$) for every function symbol f in Σ, a nonempty set $S_{\mathcal{A}} \subseteq A$ for every sort S, such that A is the union of the denotations for the sort symbols in Σ, i.e. $A = \cup \{S_{\mathcal{A}} \mid S \in S_\Sigma\}$. ■

Definition 3.2.2 Σ-Assignments
Let $\mathcal{A}$ be a Σ-quasi-algebra. We say a partial mapping $\varphi:V_\Sigma \to A$ is a **partial Σ-assignment,** iff $\varphi(x) \in S(x)_{\mathcal{A}}$ for every variable $x \in \mathcal{D}(\varphi)$. If φ is a total function, we call it a **Σ-assignment.** The **homomorphic extension** φ_h of a (partial) Σ-assignment $\varphi:V_\Sigma \to A$ on T_Σ is defined as a (partial) function $\varphi_h:T_\Sigma \to A$ as follows:

- $\varphi_h(x) := \varphi(x)$ for all Σ-variables $x \in \mathcal{D}(\varphi)$ and
- for every $f(s_1,\ldots,s_n) \in T_\Sigma$:
  if $s_i \in \mathcal{D}(\varphi_h)$ for $i = 1,\ldots,n$ and $(\varphi_h s_1,\ldots, \varphi_h s_n) \in \mathcal{D}(f_{\mathcal{A}})$
  then $f(s_1,\ldots,s_n) \in \mathcal{D}(\varphi_h)$ and $\varphi_h(f(s_1,\ldots,s_n)) := f_{\mathcal{A}}(\varphi_h s_1,\ldots, \varphi_h s_n)$. ■

A Σ-assignment assigns values to variable symbols and therefore completes the interpretation of terms in a given algebra. Thus, the corresponding homomorphic extension allows interpretation of arbitrary non-ground terms in that algebra. Now we can give the final definition for Σ-algebras.

Definition 3.2.2 Σ-Algebras
A **Σ-algebra** $\mathcal{A}$ for a signature Σ is defined as a Σ-quasi-algebra $\mathcal{A}$ that satisfies the following additional conditions:

- If $R \sqsubseteq S$ is in Σ then $R_{\mathcal{A}} \subseteq S_{\mathcal{A}}$
- For all term declarations $t{:}S \in \Sigma$ and for every partial Σ-assignment $\varphi:V_\Sigma \to A$ with $\text{Vars}(t) \subseteq \mathcal{D}(\varphi)$: $t \in \mathcal{D}(\varphi_h)$ and $\varphi_h(t) \in S_{\mathcal{A}}$. ■

The next lemma is actually the many-sorted equivalent of the well known fact that first-order terms constitute the domain (Herbrand universe) of the Herbrand interpretations.

Lemma 3.2.3 Term Algebras are Σ-Algebras
The term algebra of well-sorted terms is a Σ-algebra with carrier set T_Σ if we define:

- $S_{T\Sigma} := T_{\Sigma,S}$ for every sort $S \in S_\Sigma$.
- $\mathcal{D}(f_{T\Sigma}) := \{(s_1,\ldots,s_n) \mid f(s_1,\ldots,s_n) \in T_\Sigma\}$.
- $f_{T\Sigma}(s_1,\ldots,s_n) := f(s_1,\ldots,s_n)$. ■

Definition 3.2.4 Σ-Homomorphisms

Let Σ be a signature. A **Σ-homomorphism** is a mapping $\varphi:\mathcal{A}\to\mathcal{B}$ from a Σ-algebra $\mathcal{A}$ to a Σ-algebra $\mathcal{B}$ such that:

- $\varphi(S_{\mathcal{A}}) \subseteq S_{\mathcal{B}}$ for all $S \in S_\Sigma$.
- $\varphi(\mathcal{D}(f_{\mathcal{A}})) \subseteq \mathcal{D}(f_{\mathcal{B}})$ for all $f \in F_\Sigma$.
- If $(a_1,\ldots,a_n) \in \mathcal{D}(f_{\mathcal{A}})$ then $\varphi(f_{\mathcal{A}}(a_1,\ldots,a_n)) = f_{\mathcal{B}}(\varphi a_1,\ldots,\varphi a_n)$. ■

In particular the homomorphic extensions of variable assignments $V_\Sigma \to A$ are Σ-homomorphisms from the term algebra into the algebra $\mathcal{A}$. They will be used for the interpretation of formulae.

3.3 Semantics

A Σ-algebra does not contain objects that correspond to predicate symbols. These will be added in the definition of **Σ-structures** before we can go on and define the semantics for OSPL-formulae.

Definition 3.3.1 Σ-Structures

A **Σ-structure** $\mathcal{A}$ is a Σ-algebra which has additional denotations $P_{\mathcal{A}}$ for every predicate symbol $P \in P_\Sigma$, such that

i) $P_{\mathcal{A}}$ is a relation with $P_{\mathcal{A}} \subseteq \mathcal{A}^{\text{arity}(P)}$

ii) $=_{\mathcal{A}}$ is the identity on $\mathcal{A}$.

A Σ-homomorphism of Σ-structures $\varphi:\mathcal{A}\to\mathcal{B}$ is a Σ-homomorphism of the underlying Σ-algebras satisfying in addition $(a_1,\ldots,a_n) \in P_{\mathcal{A}} \Rightarrow (\varphi a_1,\ldots,\varphi a_n) \in P_{\mathcal{B}}$ ■

Now we can finally define the semantics of well formed formulae consisting of a **Σ-interpretation** for terms and atoms and a satisfiability relation for formulae.

Definition 3.3.2 Σ-Interpretations

Let $\mathcal{S} = (\Sigma, \mathcal{F})$ be a Σ-specification. A **Σ-interpretation** $\mathfrak{I} = (\mathcal{M}, \mathcal{V})$ for $\mathcal{F}$ is Σ-structure $\mathcal{M}$ together with a Σ-assignment $\mathcal{V}: V_\Sigma \to \mathcal{M}$.

Since T_Σ is a free Σ-structure, $\mathcal{V}$ induces a Σ-homomorphism $\mathcal{V}_h:T_\Sigma\to\mathcal{M}$. Therefore we need not distinguish between $(\mathcal{M}, \mathcal{V})$ and $(\mathcal{M}, \mathcal{V}_h)$. We use $\mathfrak{I}(t)$ as an abbreviation for $\mathcal{V}_h(t)$. ■

Notational Conventions

In the sequel we try to observe the convention to write syntactical objects with normal letters and semantical objects with italics.

$\mathfrak{I}[x/\chi]$ denotes the interpretation $\mathfrak{I}'$ that is like $\mathfrak{I}$, but maps x to χ ■

Definition 3.3.3 The Satisfiability Relation

The **satisfiability relation** ⊨ between **Σ-interpretations** and formulae is defined as follows:

Let $\mathfrak{I} = (\mathcal{M}, \mathcal{V})$ be a **Σ-interpretation**

$\mathfrak{I} \models P(t_1,\ldots,t_n)$	iff $(\mathfrak{I}(t_1),\ldots,\mathfrak{I}(t_n)) \in P_{\mathcal{M}}$
$\mathfrak{I} \models \forall x{:}S\ \mathcal{F}$	iff for all $\chi \in S_{\mathcal{M}}$: $\mathfrak{I}[x/\chi] \models \mathcal{F}$
$\mathfrak{I} \models \exists x{:}S\ \mathcal{F}$	iff there is an $\chi \in S_{\mathcal{M}}$ such that $\mathfrak{I}[x/\chi] \models \mathcal{F}$

The remaining logical connectives are interpreted as usual. ■

Summarizing we comprise the above definitions into the following definition of order-sorted predicate logic (OSPL):

Definition 3.3.4 OSPL

Following the definition scheme of def. 2.1 for logics, we define OSPL as the tuple (syntax, semantics) where syntax consists of

- ➤ the set of all sorted signatures (def. 3.1.1),
- ➤ the formation rules for well sorted terms (def. 3.1.2), i.e. the function $\Sigma \rightarrow \mathbf{T}_\Sigma$.
- ➤ the formation rules for well sorted formulae (def. 3.1.2)

and semantics consists of

- ➤ the function that maps a signature Σ to all Σ-interpretations (def. 3.3.2)
- ➤ the function that takes a Σ-interpretation $(\mathcal{M}, \mathcal{V})$ with a Σ-assignment $\mathcal{V}$ and maps it tothe Σ-interpretation $(\mathcal{M}, \mathcal{V}_h)$ where $\mathcal{V}_h$ is the induced Σ-homomorphism $\mathcal{V}_h{:}\mathrm{T}_\Sigma \rightarrow \mathcal{M}$.
- ➤ the satisfiability relation $\models$ of def. 3.3.3. ■

3.4 Quantification over Functions

In sorted predicate logics it is no problem to allow quantification over domain elements as well as over functions at the same time. If for example there is a (domain) sort D, you can introduce a sort '$D \rightarrow D$' and axiomatize this sort such that it describes really functions over D. The second-order syntax with variables in functional positions can be avoided by introducing an explicit 'apply'- function. Instead of $\forall f{:}$'$D \rightarrow D$' P(f(a)) for example one writes $\forall f{:}$'$D \rightarrow D$' P(apply(f, a)) which is first-order. With this trick you can't really encode second-order logic in first-order logic. What you loose is that a second-order quantification $\forall f{:}D \rightarrow D$... quantifies over *all* functions over D whereas a sorted first-order quantification $\forall f{:}$'$D \rightarrow D$'... quantifies only over the functions which are in the interpretation of the sort '$D \rightarrow D$'. With first-order axioms it is in general not possible to enforce that these are always all functions over D.

In Context Logic we need the "functional sorts" to quantify over context access functions, for example functions '$W \rightarrow W$' which map worlds to accessible worlds in Kripke structures. In this application it is not only not necessary to quantify over all functions $W \rightarrow W$, it would even be wrong. A second-order quantification $\forall u{:}W \rightarrow W$ ... as a replacement for the $\Box$-operator would denote not only the accessible worlds, but all worlds. Therefore with our usage of functional sorts in CL we are on the safe first-order side.

Introducing first-order functional sorts means axiomatizing the 'apply'-function and the functional composition appropriately. This can be done in the following way:

Definition 3.4.1 (Functional OSPL-Specifications)

A **functional specification** in OSPL is a specification $(\Sigma, \mathcal{F})$ consisting of a **functional signature** and the axiomatization for the two distinguished symbols $\downarrow$ (application) and $\circ$ (composition). We assume the functional sorts S_F to be a subset of the sort symbols in Σ where $\sqsubseteq_\Sigma$ is a semilattice, i.e. for each pair $S_i, S_k \in S_F$ the greatest lower bound $GLB(S_i, S_k)$ is unique if it exists. The functions we are going to define operate on the denotations of the sorts in S_F. The functional part of Σ is as follows:

1. The **functional sorts** may be '$S_1,\ldots,S_n \rightarrow^q S$', S_i and $S \in S_F$.
   Different symbols ^{q} may be used to distinguish different sets of functions $S_1 \times,\ldots,\times S_n \rightarrow S$.
   (For example in subsequent chapters we shall use '$W \rightarrow^\emptyset W$' to denote functions mapping worlds to accessible worlds in the basic accessibility relation and '$W \rightarrow^r W$' to denote functions corresponding to its reflexive closure.)

2. Whenever a declaration '$S_1,\ldots,S_n \to^q S$' $\sqsubseteq$ '$D_1,\ldots,D_n \to^r D$' $\in \Sigma$ then $D_i \sqsubseteq_\Sigma S_i$ for $i = 1,\ldots,n$ and $S \sqsubseteq_\Sigma D$ and '$S_2,\ldots,S_n \to^q S$' $\sqsubseteq$ '$D_2,\ldots,D_n \to^r D$' $\in \Sigma$ (see comment below).

3. The function declarations for $\downarrow$ are: $\downarrow$: '$S_1,\ldots,S_n \to^q S$' $\times S_1 \to$ '$S_2,\ldots,S_n \to^q S$' for every sort '$S_1,\ldots,S_n \to^q S$', and '$S_2,\ldots,S_n \to^q S$' exists as a sort symbol.
   (For our purposes we need only application to one argument at a time, i.e. we use function currying when necessary.)

4. The sort declarations for $\circ$ have the following structure:
   $\circ$:'$D_1,\ldots,D_n,S_1 \to^i S_2$' $\times$ '$E_1,\ldots,E_n,S_2 \to^j S_3$' $\to$ '$G_1,\ldots,G_n,S_1 \to^k S_3$' where $G_i := \mathrm{GLB}(D_i,E_i)$ for $i = 1,\ldots,n$, and the set of all these declarations is associative, i.e. whenever $\circ{:}F_1 \times F_2 \to F_{12}$, $\circ{:}F_{12} \times F_3 \to F_{123}$, $\circ{:}F_2 \times F_3 \to F_{23}$, $\circ{:}F_1 \times F_{23} \to F'_{123}$ are defined for functional sorts F_1, F_2 and F_3 then $F_{123} = F'_{123}$, or simply $(F_1 \times F_2) \times F_3 = F_1 \times (F_2 \times F_3)$. Furthermore the declarations are maximal. All combinations which are posssible according to these rules have to be allowed for $\circ$.
   (For the composition of m-ary functions the forst m-1 arguments are treated as parameters and only functions with the same parameters are composed, c.f. comment below.)

5. $\mathcal{F}$ contains all axioms of the following kind:
   a) $\forall f,g$:'$S_1,\ldots,S_n \to^q S$' $\quad (\forall x{:}S_1\ \downarrow(f, x) = \downarrow(g, x)) \Rightarrow f = g$.
      (Functions operating in the same way are identical.)
   b) $\forall f$:'$S_1 \to^i S_2$' $\forall g$:'$S_2 \to^j S_3$' $\forall x{:}S_1\ \ \downarrow((f \circ g), x) = \downarrow(g, \downarrow(f, x))$ (definition of composition.)
   c) Whenever $\circ$:'$D_1,\ldots,D_n,S_1 \to^i S_2$' $\times$ '$E_1,\ldots,E_n,S_2 \to^j S_3$' is defined and $n > 0$ then
      $\forall f$:'$D_1,\ldots,D_k,S_1 \to^i S_2$' $\forall g$:'$E_1,\ldots,E_k,S_2 \to^j S_3$' $\forall x_1{:}\mathrm{GLB}(D_1,E_1), \ldots, \forall x_k{:}\mathrm{GLB}(D_k,E_k)$
      $\downarrow((f \circ g), x_1,\ldots,x_k) = \downarrow(f, x_1,\ldots,x_k) \circ \downarrow(g, x_1,\ldots,x_k)$ ■

Condition 2 which reverses the sort hierarchy of functional sorts compared to the sort hierarchy of its component sorts on the domain side seems to be counterintuitive. A simple example, however, convices that this is okay. Suppose we have a sort A, the two sorts 'integer' $\sqsubseteq$ 'real' and the two functional sorts 'integer$\to$A' and 'real$\to$A'. Now, every 'real$\to$A'-function is certainly applicable to integers and is therefore also an 'integer$\to$A'-function, but not vice versa. Thus, the subsort relationship must be 'real$\to$A' $\sqsubseteq$ 'integer$\to$A'.

The composition of functions with more than one argument is asymmetrical in the arguments. The first n-1 arguments are treated different to the last argument. For example the composition of the two arithmetic functions + and * yields a function $(+ \circ *)$ with $(+ \circ *)(3, 4) = 3 * (3 + 4) = 21$. Actually to describe functions like + and * is not the intention here. The kind of n-place functions we need are essentially parametrized one-place functions, i.e. $f(x_1,\ldots,x_n) = f_{x1,\ldots,xn-1}(x_n)$, and we compose only one-place functions with the same parameter vector. The one-place functions will be used to describe general context transitions and the parameters will be used in logics with parametrized operators to enforce context switches only along labeled transitions in the context structure where the labels serve as parameters in the transition functions. The semantics of a parametrized modal operator like $\square_a$ for example is "$\square_a P$ is true in a world $\mathcal{W}$ iff P is true in all worlds which are accessible via a-labeled transitions". $\square_a P$ can therefore be translated into $\forall x$:'D,W$\to$W' $P(\downarrow(x,a))$. The 'D,W$\to$W'-functions are axiomatized such that a function $\chi \in$ 'D,W$\to$W'$_{\mathcal{C}}$ applied to $a_{\mathcal{C}}$ yields a function $W \to W$ which moves only along $a_{\mathcal{C}}$-labeled transitions. The composition of two such functions with the same parameters therefore corresponds to moves in the context structure along transitions with the same label.

In the sequel we shall use sort symbols "'S$\to$S'" as well as expressions "$S \to S$" with the usual meaning. "'S$\to$S'" in quotation marks is a syntax element and "$S \to S$" is a semantic expression. Don't mix them up.

Example: The main part of the functional OSPL-specification of MM-Logic (chapter 5) is:

1. Functional Sorts:

   $`W\to^{\emptyset}W`$, $`W\to^{r}W`$, $`W\to^{t}W`$, $`W\to^{rt}W`$, $`D,W\to^{\emptyset}W`$, $`D,W\to^{r}W`$, $`D,W\to^{t}W`$, $`D,W\to^{rt}W`$,

2. Subsort Declarations:

$$\begin{tikzcd}
 & `W\to^{rt}W' \arrow[dl, dash] \arrow[dr, dash] & \\
`W\to^{r}W' \arrow[dr, dash] & & `W\to^{t}W' \arrow[dl, dash] \\
 & `W\to^{\emptyset}W' &
\end{tikzcd}
\qquad
\begin{tikzcd}
 & `D,W\to^{rt}W' \arrow[dl, dash] \arrow[dr, dash] & \\
`D,W\to^{r}W' \arrow[dr, dash] & & `D,W\to^{t}W' \arrow[dl, dash] \\
 & `D,W\to^{\emptyset}W' &
\end{tikzcd}$$

3. Sort Declarations for $\downarrow$:

   $\downarrow$: $`D,W\to^{\emptyset}W' \times D \to `W\to^{\emptyset}W'$
   $`D,W\to^{r}W' \times D \to `W\to^{r}W'$
   $`D,W\to^{t}W' \times D \to `W\to^{t}W'$
   $`D,W\to^{rt}W' \times D \to `W\to^{rt}W'$

4. Sort Declarations for $\circ$:

   $\circ$: $`W\to^{p}W' \times `W\to^{q}W' \to `W\to^{s}W'$
   $`D,W\to^{p}W' \times `D,W\to^{q}W' \to `D,W\to^{s}W'$

   where s is derived from p and q with the following matrix:

p \ q	∅	r	t	rt
∅	t	t	t	t
r	t	rt	t	rt
t	t	t	t	t
rt	t	rt	t	rt

5. Axiomatization of $\downarrow$ and $\circ$:

   a) $\forall f,g:`W\to^{rt}W' \quad \forall x:W \downarrow(f, x) = \downarrow(g, x)) \Rightarrow f = g.$
   $\forall f,g:`D,W\to^{rt}W' \; \forall x:D \downarrow(f, x) = \downarrow(g, x)) \Rightarrow f = g$

   b) $\forall f: `W\to^{rt}W' \quad \forall g:`W\to^{q}W' \; \forall x:W \downarrow((f \circ g), x) = \downarrow(g, \downarrow(f, x))$

   c) $\forall f:`D,W\to^{rt}W' \quad \forall g:`D,W\to^{q}W' \; \forall x: D \downarrow((f \circ g), x) = \downarrow(f, x) \circ \downarrow(g, x)$ ■

Lemma 3.4.2 $\circ$ is associative in functional OSPL-specifications.

Proof: Let $f:F_f$, $g:F_g$ and $h:F_h$ where $F_f := `D_{f1},\dots,D_{fn},S_1\to^{i}S_2'$, $F_g := `D_{g1},\dots,D_{gn},S_2\to^{j}S_3'$, $F_h := `D_{h1},\dots,D_{hn},S_3\to^{k}S_4'$, and $(F_1 \times F_2) \times F_3 = F_1 \times (F_2 \times F_3)$ in the sense of 3.4.1,4 holds.

We show the associativity of $\circ$ for the two cases:

Case: $n = 0$.

Let $x:S_1$	$\downarrow((f \circ g) \circ h), x)$	$(f \circ g)$: $`S_1\to^{ij}S_3'$	
	$= \downarrow(h, \downarrow((f \circ g), x))$		(def. 3.4.1,5b)
	$= \downarrow(h, \downarrow(g, \downarrow(f, x))$	$\downarrow(f, x):S_2$	(def. 3.4.1,5b)
	$= \downarrow((g \circ h), \downarrow(f, x))$	$(g \circ h)$:$`S_2\to^{jk}S_3'$	(def. 3.4.1,5b)
	$= \downarrow((f \circ (g \circ h)), x)$		
	$\Rightarrow$ $(f \circ g) \circ h) = (f \circ (g \circ h))$		(def. 3.4.1,5a)

Case: $n>0$

Let $x_i:D_{fi}$	$\downarrow((f \circ g) \circ h), x_1,\dots,x_n)$	
	$= \downarrow((f \circ g), x_1,\dots,x_n) \circ \downarrow(h, x_1,\dots,x_n)$	(def. 3.4.1,5c)
	$= (\downarrow(f, x_1,\dots,x_n) \circ \downarrow(g, x_1,\dots,x_n)) \circ \downarrow(h, x_1,\dots,x_n)$	(def. 3.4.1,5c)
	$= \downarrow(f, x_1,\dots,x_n) \circ (\downarrow(g, x_1,\dots,x_n)) \circ \downarrow(h, x_1,\dots,x_n))$	$\downarrow(f, x_1,\dots,x_n)$:$`S_1\to^{i}S_2'$ (first case)
	$= \downarrow(f \circ (g \circ h), x_1,\dots,x_n)$	(def. 3.4.1,5c)
	$\Rightarrow$ $(f \circ g) \circ h) = (f \circ (g \circ h))$	(def. 3.4.1,5a) ■

The next theorem confirms that a functional specification really axiomatizes functions.

Theorem 3.4.3 Every model $\mathcal{A}$ for a functional specification $\mathcal{S} = (\Sigma, \mathcal{F})$ is isomorphic to a model $\mathcal{C}$ where the terms of type '$S_1,\ldots,S_n \rightarrow^q S$' are interpreted as total functions $S_{1\mathcal{C}} \times \ldots \times S_{n\mathcal{C}} \rightarrow S_{\mathcal{C}}$

Proof: Let $\mathcal{A}$ be a model for $\mathcal{S}$.

We construct $\mathcal{C}$ and the isomorphism Ξ between $\mathcal{A}$ and $\mathcal{C}$ as follows:

a) For the nonfunctional sorts S, identify $S_{\mathcal{C}}$ with $S_{\mathcal{A}}$.
   This guarantees that the subset relationships in $\mathcal{A}$ are properly transferred to $\mathcal{C}$.

b) To every $f_{\mathcal{A}} \in$ '$S_1,\ldots,S_n \rightarrow^q S$'$_{\mathcal{A}}$ we assign the function $\Xi(f_{\mathcal{A}}) = f_{\mathcal{C}}: S_{1\mathcal{C}} \times \ldots \times S_{n\mathcal{C}} \rightarrow S_{\mathcal{C}}$ which satisfies the following condition: $\forall x \in S_{1\mathcal{A}}\ f_{\mathcal{C}}(x) = \Xi(\downarrow_{\mathcal{A}}(f_{\mathcal{A}}, x))$ ($\downarrow_{\mathcal{A}}$ is the interpretation of $\downarrow$ in $\mathcal{A}$, $S_{1\mathcal{A}} = S_{1\mathcal{C}}$)
   Using the axioms 3.4.1,5a it can be shown with induction on n that $f_{\mathcal{C}}$ is unique.
   Let '$S_1,\ldots,S_n \rightarrow^q S$'$_{\mathcal{C}}$:= $\{\Xi(f_{\mathcal{A}}) \mid f_{\mathcal{A}} \in$ '$S_1,\ldots,S_n \rightarrow^q S$'$_{\mathcal{A}}\}$.
   This guarantees again that the subset relationships in $\mathcal{A}$ are properly transferred to $\mathcal{C}$. However, we have to show that the definition is consistent with the subsort relationships of the functional sorts, i.e. whenever $f_{\mathcal{A}} \in$ '$S_1,\ldots,S_n \rightarrow^q S$'$_{\mathcal{A}} \subseteq$ '$D_1,\ldots,D_n \rightarrow^r D$'$_{\mathcal{A}}$ then $\Xi(f_{\mathcal{A}})$ is a total function on $D_{1\mathcal{C}} \times \ldots \times D_{n\mathcal{C}} \rightarrow D_{\mathcal{C}}$
      Therefore let $f_{\mathcal{A}} \in$ '$S_1,\ldots,S_n \rightarrow^q S$'$_{\mathcal{A}} \subseteq$ '$D_1,\ldots,D_n \rightarrow^r D$'$_{\mathcal{A}}$ and $f_{\mathcal{C}} := \Xi(f_{\mathcal{A}})$.
      We perform induction on n.
      Base Case: $n = 1$, i.e. $f_{\mathcal{A}} \in$ '$S_1 \rightarrow^q S$'$_{\mathcal{A}}$.
      Let $x \in D_{1\mathcal{C}} = D_{1\mathcal{A}}$. According to def. 3.4.1,2, $D_1 \sqsubseteq_\Sigma S_1$, $S \sqsubseteq_\Sigma D$ and we have $D_{1\mathcal{A}} \subseteq S_{1\mathcal{A}}$.
      Thus, $x \in S_{1\mathcal{A}} = S_{1\mathcal{C}}$ and therefore $f_{\mathcal{C}}(x) = \Xi(\downarrow_{\mathcal{A}}(f_{\mathcal{A}}, x)) \in S_{\mathcal{A}} = S_{\mathcal{C}} \subseteq D_{\mathcal{C}}$. Hence, $f_{\mathcal{C}} \in D_{1\mathcal{C}} \rightarrow D_{\mathcal{C}}$
      Induction Step: $n > 1$. The induction hypothesis is
         $\forall g_{\mathcal{A}} \in$ '$S_2,\ldots,S_n \rightarrow^q S$'$_{\mathcal{A}}$: $\Xi(g_{\mathcal{A}})$ is a total function on $D_{2\mathcal{C}} \times \ldots \times D_{n\mathcal{C}} \rightarrow D_{\mathcal{C}}$
      Let again $x \in D_{1\mathcal{C}} \subseteq S_{1\mathcal{C}}$ $f_{\mathcal{C}}(x) = \Xi(\downarrow_{\mathcal{A}}(f_{\mathcal{A}}, x)) \in$ '$S_2,\ldots,S_n \rightarrow^q S$'$_{\mathcal{C}}$
      Since $\downarrow_{\mathcal{A}}(f_{\mathcal{A}}, x)) \in$ '$S_2,\ldots,S_n \rightarrow^q S$'$_{\mathcal{C}}$ according to the induction hypothesis
      $f_{\mathcal{C}}(x) = \Xi(\downarrow_{\mathcal{A}}(f_{\mathcal{A}}, x)) : D_{2\mathcal{C}} \times \ldots \times D_{n\mathcal{C}} \rightarrow D_{\mathcal{C}}$ and therefore $f_{\mathcal{C}} \in D_{1\mathcal{C}} \times D_{2\mathcal{C}} \times \ldots \times D_{n\mathcal{C}} \rightarrow D_{\mathcal{C}}$
   From a,b) we obtain that Ξ is a bijection between the sets $S_{\mathcal{A}}$ and $S_{\mathcal{C}}$ which are associated with the sorts S.

c) $\Xi(\downarrow_{\mathcal{A}}) := \downarrow_{\mathcal{C}}$ where $\downarrow_{\mathcal{C}}$ is the application function, i.e.
   $\downarrow_{\mathcal{C}}(f_{\mathcal{C}}, a_{\mathcal{C}}) = f_{\mathcal{C}}(a_{\mathcal{C}})$ holds for all $f_{\mathcal{C}}$ and corresponding arguments $a_{\mathcal{C}}$
   It is easy to verify that this definition matches the sort declarations for $\downarrow$ (def. 3.4.1,3).

d) $\Xi(\circ_{\mathcal{A}}) := \circ_{\mathcal{C}}$ where $\circ_{\mathcal{C}}$ is the composition function on the interpretations of the functional sorts.
   It is defined as follows:
   $\forall f_{\mathcal{C}} \in$ '$D_1,\ldots,D_n,S_1 \rightarrow^i S_2$'$_{\mathcal{C}}$ $g_{\mathcal{C}} \in$ '$E_1,\ldots,E_n,S_2 \rightarrow^j S_3$'$_{\mathcal{C}}$:
      $f_{\mathcal{C}} \circ_{\mathcal{C}} g_{\mathcal{C}} \in G_{1\mathcal{C}} \times \ldots \times G_{n\mathcal{C}} \times S_{1\mathcal{C}} \rightarrow S_{3\mathcal{C}}$ where $G_{i\mathcal{C}} = D_{i\mathcal{C}} \cap E_{i\mathcal{C}}$ for $i = 1,\ldots,n$ such that
      $(f_{\mathcal{C}} \circ_{\mathcal{C}} g_{\mathcal{C}})(x_1,\ldots,x_n, x) = g_{\mathcal{C}}(x_1,\ldots,x_n, f_{\mathcal{C}}(x_1,\ldots,x_n, x))$
   Using the correspondences between $f_{\mathcal{C}}$ and $f_{\mathcal{A}}$, and the fact that $\mathrm{GLB}(D_i, E_i)_{\mathcal{C}} \subseteq D_{i\mathcal{C}} \cap E_{i\mathcal{C}}$ we can prove by induction on n: $f_{\mathcal{C}} \circ_{\mathcal{C}} g_{\mathcal{C}} \in$ '$G_1,\ldots,G_n,S_1 \rightarrow^k S_3$'$_{\mathcal{C}}$.
   It is easy to verify that these definitions satisfy the axioms 3.4.1,5 and the homomorphism conditions (in both directions): $\Xi(\mathcal{D}(f_{\mathcal{A}})) = \mathcal{D}(f_{\mathcal{C}})$ and if $a \in \mathcal{D}(f_{\mathcal{A}})$ then $\Xi(f_{\mathcal{A}}(a)) = f_{\mathcal{C}}(\Xi(a))$

e) Since Ξ is a bijection between the sets $S_{\mathcal{A}}$ and $S_{\mathcal{C}}$ which are associated with the sorts S we can now assign to each other function symbol g and its corresponding function $g_{\mathcal{A}}$ a unique function $g_{\mathcal{C}} = \Xi(g_{\mathcal{A}})$ such that the homomorphism conditions hold in both directions.

f) Similarly we assign to each predicate symbol P and its corresponding relation $P_{\mathcal{A}}$ a relation $P_{\mathcal{C}} = \Xi(P_{\mathcal{A}})$ such that $(a_1,\ldots,a_n) \in P_{\mathcal{A}}$ if and only if $(\Xi(a_1),\ldots,\Xi(a_n)) \in P_{\mathcal{C}}$.

Hence, we have constructed the desired "functional" Σ-structure for the context specification which is isomorphic to $\mathcal{A}$. The isomorphism ensures that the functional Σ-structure is also a model for the specification. ■

In the sequel we shall use only the functional interpretation of functional specifications. We can exploit this interpretation also to simplify the syntax of terms, writing x(y) instead of $\downarrow$(x, y).

Chapter Four

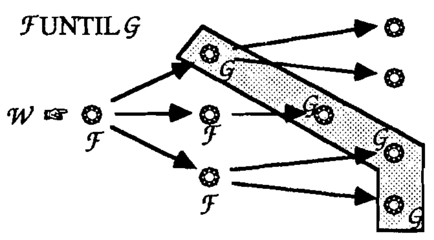
Context Logic

Context Logic (CL) consists of two parts. The first part, the "context part", is just OSPL, and this part is used to axiomatize context structures. The second part, the "domain part" with separate syntax and semantic definition is used as an intermediate language for translating specifications from the source logic into OSPL. The domain part syntax is very close to the operator syntax. For example instead of $\Box P$ in modal logic one would write $\forall x$:'W$\rightarrow$W' P in the domain syntax of CL. Translation from the operator syntax into CL is therefore almost a one to one translation where the model theoretic semantics of the operators is expressed directly with CL quantifiers. The soundness and completeness proofs for this part of the translation are therefore straightforward. The final translation from CL into OSPL is a purely syntactic transformation where the quantifications over the functional context variables are collected to build "context terms" which are attached as additional arguments to the terms and atoms. Since this is a shift of information from distributed areas of the formula tree to the term level, the soundness and completeness proofs are much more complex, but they can be done once and forall. This was actually the reason for introducing CL as an intermediate stage between logics with operator syntax and OSPL.

4.1 Syntax

A specification in CL contains both the axiomatization of the context structure, written in OSPL, and the translated formulae from some source logic. The latter ones formulate the facts about the user's domain and use context elements only occasionally as replacements for the operators. Therefore we separate the signature of CL into the context part, containing sorts and symbols concerning for example worlds and transitions between worlds, and the domain part concerning the user's sorts and symbols. The signature elements allowed in the source logic, and translated into the domain part of CL, are the usual OSPL ones, i.e. sorts, sort hierarchies, sort declarations, predicate logic quantifiers etc. For the context part, besides the OSPL elements, we need a couple of additional signature elements. To motivate the additional elements, let us consider the representation of an 'until' operator for branching time temporal logic. Its semantics is:

$\mathcal{F}$ UNTIL $\mathcal{G}$ holds in a world $\mathcal{W}$ iff on each path $\mathcal{P}$ in the possible worlds structure starting with $\mathcal{W}$, $\mathcal{G}$ holds eventually in a world $\mathcal{W}_1$ on $\mathcal{P}$ and $\mathcal{F}$ holds in all worlds on $\mathcal{P}$ between $\mathcal{W}$ and $\mathcal{W}_1$.
Graphically:



First of all we notice that, besides the concept of worlds, we need the concept of *paths* through the possible worlds structure; and we need transitions from worlds to paths and back to worlds. Transitions from worlds to paths can be described with **context access functions** mapping worlds to paths, and transitions from paths to worlds can be described with functions mapping paths to worlds. A slightly more complex example would show that we need also transitions from a world on a path to another world on *the same path.* Functions representing these transitions need both, a world and a path as input. The real context information that is necessary to handle modal operators together with an 'until' operator are therefore tuples <world,path>. Since, however, terms and atoms are interpreted only in worlds and the path component becomes irrelevant, we finally must project the world component out of the tuples. Thus, the additional components we need in the context part of the CL signature are:

- a separation of the **context sorts** (W, WP in the example) into
  - the **basic context sorts** (the sort WP describing tuples <world,path>) and
  - the **interpretation context sorts** (the sort W), the sorts which are relevant for the interpretation of terms and atoms.
- a **symbol variation function** that assigns to each function, predicate and domain variable symbol the projector functions mapping the actual context (WP) to the interpretation context (W). We need this information for domain variable symbols too because variables of existential force are Skolemized, i.e. turned into function symbols.

Of course we don´t restrict the context information to one single context, but arbitrary many contexts, like for example time and belief, are allowed to operate independently from each other. (Worlds and paths in the above example are not independent from each other.)

Definition 4.1.1 CL-Signatures

CL-signatures are essentially functional OSPL-signatures (c.f. def. 3.4.1) with a few more concepts:

1. We separate the symbols into two partially overlapping parts, the **context signature** which contains the functional part we are interested in and the **domain signature** according to the following criteria:
   i) The sort symbols consist of three nonempty and disjoint parts:
      a) the **domain sorts** which belong to the domain signature;
      b) the **context sorts** which belong to the context signature;
         a subset of the context sorts is selected as **basic context sorts**,
         (in the sequel we assume a fixed ordering of the basic context sorts) and
      c) the **functional context sorts** which also belong to the context signature.
         The functional context sorts are of type '$C \rightarrow^{i} C$' or '$D,C \rightarrow^{i} C$' where C is a basic context sort and D is a domain sort.
      d) The domain sorts occurring in the functional context sorts belong to both parts of the signature.
   ii) **Context symbols**, i.e. variable, function and predicate symbols with context sorts in their sort declarations belong to the context signature, the **domain symbols**, i.e. the symbols with only domain sorts in their sort declarations belong to the domain signature.
       The equality symbol belongs to both parts of the signature.
       A variable of sort '$C \rightarrow^{i} C$' or '$D,C \rightarrow^{i} C$' is called a **functional context variable**.

2. The sort declaration are such that
   i) In the subset '$C \rightarrow^{q} C$' of functional context sorts (with different ^{q}) there is always a unique topsort.
   ii) Except for the equality symbol there are no context predicates with functional context sorts in their sort declaration.
3. A CL-signature contains two more objects:
   i) The **interpretation context sorts** is a fixed tuple $I_1,\ldots,I_k$ of context sorts.
   ii) A **symbol variation function** $\mathcal{SV}$ mapping each domain symbol to a tuple $(p_1,\ldots,p_k)$ of one-place function symbols $p_i{:}C_i \rightarrow I_i$ with basic context sorts as domainsort and interpretation context sort as rangesort, I_i being the i-th element in the interpretation context sorts. ■

Definition 4.1.2 CL-Terms

The set of CL-terms over a CL-signature Σ is simply the set T_Σ of OSPL-terms (def. 3.1.2).
Domain terms are terms built from domain symbols only, all other terms are **context terms**.
Terms of sort '$C \rightarrow^{q} C$' or '$D,C \rightarrow^{q} C$' are called **context access terms**. ■

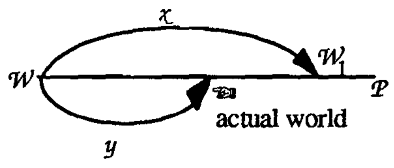
The syntax rules for CL-terms need not be separated into rules for domain terms and context terms, OSPL-syntax is good for both. For the formula syntax, however, we need special formation rules for domain formulae. First of all we may have quantifications like $\forall x$:'W→W' P, where the quantified variable does not occur in the body of the quantification. This, of course, is also possible in predicate logic syntax, but there it makes not much sense. You can drop the quantifier without any logical consequences. In Context Logic, these quantifiers are the main constituents because nested quantifications of this type accumulate the context information for the interpretation of the terms and atoms. Dropping them changes the logical status of a formula considerably.

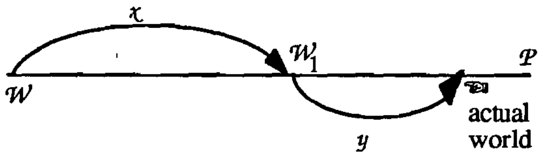
For special purposes, variations of the quantifications over context access functions are needed. Sometimes it is necessary to unwind the context information to the state of an embracing quantifier. This occurs for example in the translation of an 'until' operator (def. 5.2.2):

$$\Psi(\mathcal{F}\,\mathrm{UNTIL}\ \mathcal{G}) = \forall p{:}\text{'W}\to\text{P'}\ \exists x{:}\text{'P}\to\text{W'}\ (\Psi(\mathcal{G}) \wedge \forall y{:}\text{'P}\to\text{W'-x}\ \ p\circ y < p\circ x \Rightarrow \Psi(\mathcal{F}))$$

which is a straightforward encoding of the semantics of 'until' as given above. It should be read: From the initial world, say $\mathcal{W}_0$, obtain a path $\mathcal{P} = p(\mathcal{W}_0)$, starting with $\mathcal{W}_0$ by applying a "world-to- path-function" p to $\mathcal{W}_0$. $\mathcal{W}_0$ is still the actual world context. Then move from $\mathcal{W}_0$ to $\mathcal{W}_1 = x(\mathcal{W}_0, \mathcal{P})$ *on the path* $\mathcal{P}$ by applying the a "path-to-world-function" x that exists according to the $\exists x$:'P→W'-quantification. Evaluate $\Psi(\mathcal{G})$ in the world $\mathcal{W}_1$, which is now the actual world context. Now reckon a world $\mathcal{W}_2 = y(\mathcal{W}_0, \mathcal{P})$ on $\mathcal{P}$ by applying "path-to-world-function" y, not to the actual world context $\mathcal{W}_1$, as the nesting of the quantifiers normally suggests, but, because of the "-x" appendix in the $\forall y$:'P→W'-x quantification, to the world $\mathcal{W}_0$ which is the state of the world-context before the $\exists x$:'P→W' quantification. Evaluate "<(p∘y, p∘x) ⇒ $\Psi(\mathcal{F})$" in the world $\mathcal{W}_2$. Hence, in the quantification $\forall y$:'P→W'-x, the "-x" appendix enforces the context of y to be unwound to the state before the quantification of x, thus making x invisible for y and therefore the worlds denoted by x and y independent from each other. Pictorially, the difference between the formula with and without the "-x" appendix is:

$\forall p$:'W→P' $\exists x$:'P→W' $\forall y$:'P→W'-x $\qquad\qquad$ $\forall p$:'W→P' $\exists x$:'P→W' $\forall y$:'P→W'





Consequently, the first modification on CL-quantifiers is to allow a "-v" appendix where v is a context variable in the scope of a more outside quantification.

The second modification is aimed to provide a representation for indexed operators. Indexed operators like for example $\Box_s$ (we write it $[\![s]\!]$) have been used in epistemic and action logics where the index refers to an agent or an action respectively. For example in an "belief interpretation" the formula $[\![\text{Tom}]\!]$ loves(John, Mary) means "Tom believes that John loves Mary" or technically "in all worlds which are compatible with Tom's current knowledge, John loves Mary". The semantics of the indexed operators can be given either by associating with each index a separate accessibility relation or, and that is only another way to say it, by labeling the transitions in the accessibility relation with the indices themselves or interpretations of the indices. $\Box_s$P is now interpreted: P holds in all worlds accessible via s-labeled transitions. The representation of indexed operators in CL requires context access functions which are parametrized with the labels. When we choose the indices to be arbitrary terms and the labels to be the interpretation of the index terms, we can represent the parametrized context access functions as 'D,W→W'-functions where D is the domain sort used for the index terms. A quantification "for all s-indexed transitions" could now be represented as $\forall\downarrow$(x:'D,W→W', s) expressing that the quantification ranges over all 'D,W→W' functions, but the actual world is to be obtained by applying the interpretation of x, the parametrized world access function, to the interpretation of s, the label.

A technical difficulty arises with the interpretation of the parameter "s". Although usually a quantified context variable in CL occurs only once, in the corresponding quantification, there are operators like the parametrized eventually operator (def. 5.2.2) whose translation has more occurrences of parametrized context variables. The parameter, i.e. the "s" in "$\forall\downarrow(x, s)$" is uniquely bound to the variable x itself and therefore x should only occur in the term "$\downarrow(x, s)$". "s" has to be interpreted in the context of the quantifier where it occurs. A second occurrence, however, might be in the scope of another operator and the term "s" would then be interpreted differently to its first occurrence. To avoid this, we write instead of "$\forall\downarrow(x, s)$" "$\forall\downarrow(x, z{=}s)$" where z is a variable of the same sort of s. And instead of "$\downarrow(x, s)$" in other parts of the formula we write "$\downarrow(x, z)$". Since we assume constant-domain interpretations, its interpretation is in all worlds the same and the definition of the satisfiability relation below ensures that the interpretation of z equals the interpretation of s.

A quantification over a context access variable shifts as a side effect the actual context to the context obtained after application of the context access function associated with the variable to the actual context. Sometimes it is necessary to have only this side effect, generated by a term t instead of a variable. Therefore we introduce a new operator $\wp$ which is applied to a context access term and a formula. $\wp$ t $\mathcal{F}$ means that $\mathcal{F}$ is to be interpreted in the context obtained after application of the t's interpretation, which is a context access function, to the actual context. Thus $\wp$ denotes a simple context shift, nothing else. If for example t is "tomorrow" and $\mathcal{F}$ is "bad-weather" then "tomorrow" can be interpreted as a constant temporal context shift and "$\wp$ tomorrow bad-weather" means simply that there will be bad weather tomorrow.

There is yet another special feature of the domain formulae in CL that should be mentioned. As can be seen in the translation rule of the 'until' operator above, special predicates like the <-predicate operating on contexts only may be necessary to formulate the translation rules. These predicates cannot be used in the source logic. Although after the final translation into OSPL, the predicates really operate on contexts (worlds), in the CL stage of the translation their arguments must be context access terms because the actual context is not yet known. That's the reason for the somewhat peculiar sort declaration for these predicates. They are defined with context sorts, but used with context access terms.

Definition 4.1.3 CL-Formulae

We distinguish two types of formulae, the **domain formulae** and the **context formulae**. The context formulae over a signature Σ are simply the set of well formed OSPL-formulae (def. 3.1.2) over the context part of the signature. The domain formulae are built from **domain atoms** and **domain literals** as follows:

Domain atoms are the least set such that:

(i) Every well formed OSPL atom built with a domain predicate, except the equality symbol, is a domain atom. Atoms built with the equality symbol and domain terms are also domain atoms.

(ii) Atoms built with a context predicate P are also domain atoms if they are built according to the usual rules, however after having replaced context sorts C with '$C\to C$' in the sort declaration for P, '$C\to C$' being the topsort in the sublattice of all '$C\to^{q}C$' sorts.That means instead of terms of sort C we require terms of sort '$C\to C$' as arguments of P.

Domain literals are domain atoms or negated domain atoms.

In the sequel $\pm L$ denotes either a positive literal, i.e. L itself, or a negative literal, i.e. $\neg L$.

The set of variables occurring as subterms in an atom are its **free variables**.

The set of domain formulae over the given signature is defined as the least set such that

(iii) domain atoms and domain literals are domain formulae.

(iv) If $\mathcal{F}$ is a domain formula, x:'$D,C\to^{q}C$' is a context access variable, z:D is a domain variable and s is a domain term of sort D and x and z occur only as subterms of $\downarrow(x, z)$ in $\mathcal{F}$ then $\mathcal{G} = \forall\downarrow(x, z{=}s)\mathcal{F}$ and $\mathcal{G} = \exists\downarrow(x, z{=}s)\ \mathcal{F}$ are domain formulae.
If V is the set of free variables of $\mathcal{F}$ then $V\setminus\{x,z\} \cup \mathrm{Vars}(s)$ is the set of free variables of $\mathcal{G}$.

(v) If $\mathcal{F}$ and $\mathcal{G}$ are domain formulae, x is a domain variable or a context variable of sort '$C\to^{q}C$' then $\neg\mathcal{F}$, $\mathcal{F}\wedge\mathcal{G}$, $\mathcal{F}\vee\mathcal{G}$, $\forall x\ \mathcal{F}$ and $\exists x\ \mathcal{F}$ are domain formulae.
If V is the set of free variables of $\mathcal{F}$ then $V\setminus\{x\}$ is the set of free variables of $\forall x\ \mathcal{F}$ and $\exists x\ \mathcal{F}$.

(vi) If $\forall t\ \mathcal{F}$ or $\exists t\ \mathcal{F}$ is a domain formula where $t = x$:'C$\rightarrow^q$C' or $t = \downarrow$(x:'D,C$\rightarrow^q$C',z=s) and y is a context variable of sort 'D,C$\rightarrow^q$C' or 'C$\rightarrow^q$C' then $\mathcal{G} = \forall$t-y $\mathcal{F}$ and $\mathcal{G} = \exists$t-y $\mathcal{F}$ are domain formulae. The free variables for $\mathcal{G}$ are those of $\forall t\ \mathcal{F}$ plus the variable y.

(vii) If $\mathcal{F}$ is a domain formula and t is a context access term of sort 'C$\rightarrow$C' for some C then $\wp$ t $\mathcal{F}$ is a domain formula.

As usual we assume that formulae are standardized apart, i.e. formulae like $\forall x \exists x$... do not occur. ■

In the sequel we assume that CL-specifications are always functional specifications according to def. 3.4.1 and def. 4.1.1 where the functional part consists of the context part of the signature, and the axioms are formulated with context formulae. The remaining part of the axioms is formulated with domain formulae. We call the context part of a CL-specification the **context specification.**

Examples for domain formulae:

$\forall$x:'W$\rightarrow$W' P

$\exists\downarrow$(x:'D,W$\rightarrow$W', z = a) P

$\forall$x:'W$\rightarrow$W $\wp$(BF(t) $\circ$ +1) Q

$\forall$p:'W$\rightarrow^1$W' $\exists$x:'W$\rightarrow^2$W' (P $\wedge$ $\forall$y:'W$\rightarrow^2$W'-x (<(p$\circ$y, p$\circ$x) $\Rightarrow$ Q)). ■

4.2 Semantics

The semantics of CL is essentially a Kripke style possible worlds semantics [Kripke 59,63], its technical formulation, however, is adapted to the necessities of CL. The worlds are determined by the interpretation of the interpretation context sorts. If $C_1,\ldots,C_n$, for example, are the interpretation context sorts, then tuples $(c_1,\ldots,c_n)$ of interpretations of $C_1,\ldots,C_n$ denote the worlds. (In modal logics these tuples are always singletons). We assign a Σ-structure, i.e. a predicate logic interpretation to each of these tuples. The accessibility relation is determined by the interpretation of the 'C$\rightarrow$C'-sorts. Since they are interpreted as functions one can think of function applications as transitions from world to world and of the argument-value relation as the accessibility relation. The application of a 'D,C$\rightarrow$C'-sort to a domain element yields a normal 'C$\rightarrow$C'-function, therefore 'D,C$\rightarrow$C'-functions are used to represent labeled transitions.

Definition 4.2.1 (Frames)

A *frame* for a CL-signature consisting of the context signature Σ_C and the domain signature Σ_D and interpretation context sorts $(I_1,\ldots,I_n)$ is a tuple $(\mathcal{C}, \mathcal{SV})$ where

$\mathcal{C}$ is a functional Σ_C-structure (this is actually the possible worlds structure).

$\mathcal{SV}$ is a function that assigns to each tuple $(i_{1\mathcal{C}},\ldots,i_{n\mathcal{C}}) \in I_{1\mathcal{C}} \times\ldots\times I_{n\mathcal{C}}$ of **contexts** a Σ_D-structure $\mathcal{A}_{i1\ldots in}$ where the domains are identical in all these Σ_D-structures (denoted as $D_{\mathcal{SV}}$ for a sort D) and $D_{\mathcal{SV}} = D_{\mathcal{C}}$ for each domain sort D belonging to Σ_C and Σ_D.
(Each world is viewed as an interpretation in the predicate logic sense. The constant domain assumption is built in.) ■

Notice that we use the notion *frame* somewhat different to [Fitting 83]. A frame in the definition above includes already the interpretation of the nonlogical symbols. The functional Σ_C-structure $\mathcal{C}$ corresponds to Fittings frame.

Definition 4.2.2 (Signature Interpretations)
A signature interpretation for a signature Σ with basic context sorts $(C_1,\ldots,C_m)$ is a tuple $(\mathcal{F}, \mathcal{V}, \mathcal{C}, \mathcal{P})$ where

$\mathcal{F} =: (\mathcal{C}, \mathcal{SV})$ is a frame,

$\mathcal{V}$ is a Σ-assignment, the **actual assignment** and

$\mathcal{C}$ is a tuple $(c_1,\ldots,c_m) \in C_{1\mathcal{C}} \times \ldots \times C_{m\mathcal{C}}$ of actual contexts and

$\mathcal{P}$ is a Σ-assignment, the **context assignment.** ■

The $\mathcal{V}$-component is the usual assignment of values to variables, domain elements to domain variables and context access functions to context access variables. $\mathcal{V}$ is irrelevant for the interpretation of closed formulae. It is used by the satisfiability relation (see below) in the usual way for recording variable bindings during the recursive descent into a formula. The $\mathcal{C}$-component contains the actual context (actual world) in which the formula or term has to be interpreted. For a closed formula it gives the initial context (initial world) for its interpretation, and during the recursive descent into the formula the satisfiability relation uses it to record in a stack-like manner the actual context. The actual context changes each time a new quantifier over a functional context variable is encountered. The $\mathcal{V}$ and $\mathcal{C}$-components are standard for modal logics. The $\mathcal{P}$-component, however, is specific for CL. It records for each functional context variable the context of its quantification, i.e. its first occurrence. This information is necessary for unwinding the $\mathcal{C}$-context when a quantifier with "-y" appendix is encountered and for the proper interpretation of the special context predicates (like the <-predicate above).

The actual Σ-structure (world) to be used for the interpretation of terms and atoms is obtained form the actual context, i.e. the $\mathcal{C}$-component of the interpretation by applying the topsymbol's projector functions to the actual context. For example in the multi modal logic we describe in chapter 5, the actual context consists of tuples <world, path>, but the actual Σ-structure for the interpretation of a term is independent of the path component. Therefore the project-world function has to be applied to the actual context in order to get the interpretation context, i.e. the actual Σ-structure.

Definition 4.2.3 (The Actual Σ-Structure)
Given a signature interpretation $((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ for a signature Σ with basic context sorts $C_1,\ldots,C_m$, interpretation context sorts $I_1,\ldots,I_k$ and symbol variation function $\mathcal{SV}$, we define the **actual Σ-structure** for a function or predicate symbol f as $\mathcal{SV}(i_{1\mathcal{C}},\ldots,i_{k\mathcal{C}})$ where the $i_{\mathcal{C}}$ as elements of the Σ-Structure $\mathcal{C}$ are determined from $\mathcal{SV}(f) = (p_1,\ldots, p_k)$ and $\mathcal{C}$ as follows:
If p_i is a function $p_i{:}C_i \to I_i$ then $i_{\mathcal{C}} := p_{i\mathcal{C}}(c_C)$ where c_C is the element in $\mathcal{C}$ belonging to the context sort C. ■

Definition 4.2.4 (Interpretation of Terms)
Given a signature interpretation $\mathfrak{I} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ for a signature Σ, context terms are interpreted by the corresponding induced homomorphism $\mathcal{V}_h$ from the algebra of context terms into $\mathcal{C}$. (That is the same as in OSPL.) For interpreting domain terms we turn $\mathfrak{I}$ into an homomorphism $\mathfrak{I}_h$ from the algebra of domain terms into the actual Σ-structure of the term's topsymbol.

$\mathfrak{I}_h(x) := \mathcal{V}(x)$ where x is a variable symbol,

$\mathfrak{I}_h(f(t_1,\ldots,t_n)) := f_{\mathcal{A}}(\mathfrak{I}_h(t_1),\ldots,\mathfrak{I}_h(t_n))$ where $\mathcal{A}$ is the actual Σ-structure of f (def. 4.2.3).

In the sequel we do not distinguish between $\mathfrak{I}$, $\mathfrak{I}_h$ and $\mathcal{V}_h$. ■

Notational Conventions:

Let $\mathfrak{I} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ be a signature interpretation.

$\mathcal{C}_C$ denotes the element of $\mathcal{C}$ that corresponds to the sort C.

In general it should be clear which $\mathcal{C}$, i.e. which $\mathfrak{I}$ is meant.

$\mathfrak{I}[x/\chi]_{\mathcal{V}}$ denotes the interpretation $\mathfrak{I}'$ where $\mathcal{V}[x/\chi]$ maps x to χ and which is otherwise like $\mathfrak{I}$.

$\mathfrak{I}[x/c]_{\mathcal{P}}$ denotes the interpretation $\mathfrak{I}'$ where $\mathcal{P}[x/c]$ maps x to c and which is otherwise like $\mathfrak{I}$.

$\mathfrak{I}[C/c]_{\mathcal{C}}$ denotes the interpretation $\mathfrak{I}'$ where in $\mathcal{C}$ the element belonging to the sort C is replaced by c, and which is otherwise like $\mathfrak{I}$.

We use combinations of these notations with the corresponding meaning. ■

Definition 4.2.5 (The Satisfiability Relations)

Let $\mathfrak{I} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ be a signature interpretation for a signature Σ.

The satisfiability relation for context formulae is just the OSPL satisfiability relation where all but $\mathcal{C}$ and $\mathcal{V}$ is ignored in $\mathfrak{I}$.

The satisfiability relation for domain formulae is defined as follows:

$\mathfrak{I} \Vdash P(t_1,\dots,t_n)$ iff $(\mathfrak{I}(t_1),\dots,\mathfrak{I}(t_n)) \in P_{\mathcal{A}}$ where $\mathcal{A}$ is the actual Σ-structure of P and P is a domain predicate (def. 4.2.3)

$\mathfrak{I} \Vdash P(t_1,\dots,t_n)$ iff $(t_1,\dots,t_n) \in P_{\mathcal{C}}$, where P is a context predicate and the t_i are determined as follows:

If t_i is a context variable x or a term $\downarrow(x,z)$ then $t_i := \mathfrak{I}(t_i)(\mathcal{P}(x))$

If $t_i = s \circ t$ and s is a context variable x or a term $\downarrow(x,z)$ then $t_i := \mathfrak{I}(t)(\mathfrak{I}(s)(\mathcal{P}(x)))$

otherwise $t_i := \mathfrak{I}(t_i)$

$\mathfrak{I} \Vdash \neg\mathcal{F}$ iff not $\mathfrak{I} \Vdash \mathcal{F}$

$\mathfrak{I} \Vdash \mathcal{F} \wedge \mathcal{G}$ iff $\mathfrak{I} \Vdash \mathcal{F}$ and $\mathfrak{I} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \mathcal{F} \vee \mathcal{G}$ iff $\mathfrak{I} \Vdash \mathcal{F}$ or $\mathfrak{I} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \forall x\, \mathcal{G}$ where x is a domain variable of sort D
 iff for every $\chi \in D_{\mathcal{SV}}$: $\mathfrak{I}[x/\chi]_{\mathcal{V}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \forall x\, \mathcal{G}$ where x is a context variable of sort '$C \rightarrow^{q} C$'
 iff for every $\chi \in$ '$C \rightarrow^{q} C$'$_{\mathcal{C}}$: $\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \forall x\text{-}y\, \mathcal{G}$ where x is a context variable of sort '$C \rightarrow^{q} C$' and y is a variable of sort '$C \rightarrow^{P} C$' or '$D,C \rightarrow^{P} C$'
 iff for every $\chi \in$ '$C \rightarrow^{q} C$'$_{\mathcal{C}}$: $\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \forall{\downarrow}(x,z=t)\, \mathcal{G}$ where x is a context variable of sort '$D,C \rightarrow^{q} C$'
 iff for every $\chi \in$ '$D,C \rightarrow^{q} C$'$_{\mathcal{C}}$: $\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \forall{\downarrow}(x,z=t)\text{-}y\, \mathcal{G}$ where x is a context variable of sort '$D,C \rightarrow^{q} C$'
 iff for every $\chi \in$ '$D,C \rightarrow^{q} C$'$_{\mathcal{C}}$: $\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \exists x\, \mathcal{G}$ where x is a domain variable of sort D
 iff there is an $\chi \in D_{\mathcal{SV}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \exists x\, \mathcal{G}$ where x is a context variable of sort '$C \rightarrow^{q} C$'
 iff there is an $\chi \in$ '$C \rightarrow^{q} C$'$_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \exists x\text{-}y\, \mathcal{G}$ where x is a context variable of sort '$C \rightarrow^{q} C$' and y is a variable of sort '$C \rightarrow^{P} C$' or '$D,C \rightarrow^{P} C$'
 iff there is an $\chi \in$ '$C \rightarrow^{q} C$'$_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \exists{\downarrow}(x,z=t)\, \mathcal{G}$ where x is a context variable of sort '$D,C \rightarrow^{q} C$'
 iff there is an $\chi \in$ '$D,C \rightarrow^{q} C$'$_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \exists{\downarrow}(x,z=t)\text{-}y\, \mathcal{G}$ where x is a context variable of sort '$D,C \rightarrow^{q} C$'
 iff there is an $\chi \in$ '$D,C \rightarrow^{q} C$'$_{\mathcal{C}}$: $\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \Vdash \mathcal{G}$

$\mathfrak{I} \Vdash \wp\, t\, \mathcal{G}$ where t is a term of sort '$C \rightarrow^{q} C$'
 iff $\mathfrak{I}[C/\mathfrak{I}(t)(\mathcal{C}_C)]_{\mathcal{C}} \Vdash \mathcal{G}$ ■

Notice that in quantifications $\forall{\downarrow}(x,z=t)\text{-}y\ \mathcal{G}$ and $\exists{\downarrow}(x,z=t)\text{-}y\ \mathcal{G}$ the term 't' is interpreted in the actual context, i.e. the "-y" suffix has no effect on 't'.

Lemma 4.2.6 (Negation Normal Form)

The following transformation rules for domain formulae can now be proved being equivalence preserving:

$$\neg(\mathcal{F}\wedge\mathcal{H}) \to \neg\mathcal{F}\vee\neg\mathcal{G} \qquad \neg(\mathcal{F}\vee\mathcal{H}) \to \neg\mathcal{F}\wedge\neg\mathcal{G}$$

$$\neg\forall x\ \mathcal{G} \to \exists x\ \neg\mathcal{G} \qquad \neg\exists x\ \mathcal{G} \to \forall x\ \neg\mathcal{G}$$

$$\neg\forall x\text{-}y\ \mathcal{G} \to \exists x\text{-}y\ \neg\mathcal{G} \qquad \neg\exists x\text{-}y\ \mathcal{G} \to \forall x\text{-}y\ \neg\mathcal{G}$$

$$\neg\forall{\downarrow}(x,z=t)\ \mathcal{G} \to \exists{\downarrow}(x,z=t)\ \neg\mathcal{G} \qquad \neg\exists x{\downarrow}(x,z=t)\ \mathcal{G} \to \forall{\downarrow}(x,z=t)\ \neg\mathcal{G}$$

$$\neg\forall{\downarrow}(x,z=t)\text{-}y\ \mathcal{G} \to \exists{\downarrow}(x,z=t)\text{-}y\ \neg\mathcal{G} \qquad \neg\exists x{\downarrow}(x,z=t)\text{-}y\ \mathcal{G} \to \forall{\downarrow}(x,z=t)\text{-}y\ \neg\mathcal{G}$$

$$\neg\wp\ t\ \mathcal{G} \to \wp\ t\ \neg\mathcal{G}$$

These rules allow to move negations in front of the atoms, i.e. to produce a negation normal form. ∎

To illustrate the last rule, assume t means again "tomorrow", i.e. a fixed temporal context shift and $\mathcal{G}$ means "bad-weather". Then $\neg\ \wp$ tomorrow bad-weather $\to \wp$ tomorrow $\neg$bad-weather means that 'not tomorrow is bad weather' is equivalent with 'tomorrow is not bad weather'. Sounds reasonable, doesn't it.

Assumption: In the sequel we always assume that domain formulae are in negation normal form.

4.3 A Logic Morphism from Context Logic to Order-Sorted Predicate Logic.

We define a logic morphism $\Pi = (\Pi_{\mathcal{S}}, \Pi_{\mathfrak{I}})$ (def. 2.3) from CL to OSPL that allows to translate CL-specifications into OSPL-specifications, thus enabling proofs by translation and refutation. For the specification morphism $\Pi_{\mathcal{S}}$ we define a signature morphism Π_Σ and a formula morphism $\Pi_{\mathcal{F}}$ as components first.

The main task for the signature morphism is to provide additional "context arguments" for the domain function and predicate symbols. The sort declarations must be extended for this purpose.

Definition 4.3.1 The Signature Morphism Π_Σ

A CL-signature Σ with symbol variation function $\mathcal{SV}$ is mapped to an OSPL-signature as follows:

1. The sorts, the sort hierarchy and the context symbols, including the equality symbol, remain unchanged.
2. For each basic context sort C a distinguished constant symbol 0_C:C is introduced.
3. Each n-place domain function or predicate symbol f is mapped to an k+n-place function or predicate symbol where k is the number of interpretation context symbols.
   (The k additional arguments take the context terms for the functional context sorts.)
4. In the term declarations t:S the term t is modified by the following recursive function π_Σ:
   $\pi_\Sigma(x) = x$ where x is a variable.
   $\pi_\Sigma(f(t_1,\ldots,t_n)) = \Pi_\Sigma(f)(y_1,\ldots,y_k, \pi_\Sigma(t_1),\ldots,\pi_\Sigma(t_1))$ (❖)
   where the y_i are new variables of sort I_i such that $(I_1,\ldots,I_k)$ is just the tuple of rangesorts of the functions in $\mathcal{SV}(f)$ (which equals the interpretation context sorts).
   If f is a context symbol then k = 0.
5. The predicate declarations are modified according to the rule (❖).
6. A sufficiently large set of function symbols is added to serve as Skolem functions. ∎

The formula morphism translates domain formulae into OSPL-formulae by collecting the quantifications over context access functions and inserting corresponding context terms as additional arguments into the terms and atoms.

Definition 4.3.2 The Formula Morphism $\Pi_{\mathcal{F}}$
Since context formulae are already OSPL-formulae, $\Pi_{\mathcal{F}}$ needs only translate the domain formulae into OSPL-formula. $\Pi_{\mathcal{F}}$ requires an auxiliary function π that makes the recursive descent into the formulae and terms to be translated. π has two additional arguments $\mathbb{c}$ and $\mathbb{p}$. $\mathbb{c}$ accumulates for each basic context sort C the sequence of nested quantifiers over context functions, i.e. when $(C_1,\ldots,C_n)$ is the tuple of basic context sorts, $\mathbb{c}$ is a tuple $(t_1,\ldots,t_n)$ of terms where the t_i correspond to the sorts C_i. t_i is a '$C_i \rightarrow {}^{q}C_i$'-term of the structure $t_{i1}\circ\ldots\circ t_{im}$. The t_{ij} are either variables or terms $\downarrow(x, t)$. $\mathbb{p}$ records for each context variable of sort '$C_i \rightarrow {}^{q}C_i$' or '$D,C_i \rightarrow {}^{q}C_i$' the C-context of the corresponding quantification, i.e. the value, $\mathbb{c}_C$ had when the quantification that introduced x was translated.

To simplify notation we assume that the initial value of $\mathbb{c}$ is a tuple of identity functions and in $\mathbb{p}$ the assignment for each variable is the also the identity-function (which of course should be eliminated in a real implementation). For example when the formula $\forall x$:'$C \rightarrow C$' $\forall y$:'$C \rightarrow C$' P is translated, at the time when π arrives at P, $\mathbb{c} = (x\circ y)$, $\mathbb{p}(x)$ = identity-function and $\mathbb{p}(y) = x$. (Since $\mathbb{p}$ records terms, a Lisp hacker may think of $\mathbb{p}$ as an association list for variable symbols.)

Notational Conventions:

$\mathbb{c}_C$ denotes the element of $\mathbb{c}$ that corresponds to the sort C.
$\mathbb{c}[C/c]$ is like $\mathbb{c}$ except that the element belonging to the sort C is replaced by c.
$\mathbb{p}[x/c]$ is like $\mathbb{p}$ except that x is mapped to c.
$\pi(\mathcal{G}\ldots)[x\leftarrow t]$ means "translate $\mathcal{G}$ and afterwards replace all occurrences of x by t".

$\pi(\mathcal{F}, \mathbb{c}, \mathbb{p})$ is defined inductively over the structure of $\mathcal{F}$:

A) $\pi(x, \mathbb{c}, \mathbb{p}) := x$ where x is any variable symbol

B) $\pi(f(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}) := \Pi_\Sigma(f)(s_1,\ldots,s_k, \pi(t_1, \mathbb{c}, \mathbb{p}),\ldots, \pi(t_n, \mathbb{c}, \mathbb{p}))$ where f is a function symbol.
The s_i are determined as follows:
If f is a context function then $k = 0$.
Otherwise $s_i := p_i(\downarrow(\mathbb{c}_{Ci}, 0_{Ci}))$ where $p_i{:}C_i \rightarrow I_i$ is the i-th projector function in $\mathcal{SV}(f)$.

C) $\pi(\pm P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}) := \pm\Pi_\Sigma(P)(s_1,\ldots,s_k, \pi(t_1, \mathbb{c}, \mathbb{p}),\ldots, \pi(t_n, \mathbb{c}, \mathbb{p}))$ where P is a domain predicate.
The s_i are determined as in the previous case. ("$\pm$" means negated or unnegated.)

D) $\pi(\pm P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}) := \pm P(s_1,\ldots,s_n)$ where P is a context predicate.
The s_i are determined as follows:
If t_i:'$C \rightarrow {}^{q}C$' is either a context variable x or a term $\downarrow(x,z)$ then $s_i := \downarrow(\mathbb{p}(x)\circ t_i, 0_C)$.
If $t_i = s \circ t$ and s is a context variable x:'$C \rightarrow {}^{q}C$' or a term $\downarrow(x,z)$ then $s_i := \downarrow(\mathbb{p}(x)\circ s\circ\pi(t, \mathbb{c}, \mathbb{p}), 0_C)$,
otherwise $s_i := \pi(t_i, \mathbb{c}, \mathbb{p})$.

E) $\pi(\forall x\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}, \mathbb{p})$ where x is a domain variable.

F) $\pi(\forall x\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{c}_C\circ x], \mathbb{p}[x/\mathbb{c}_C])$ where x:'$C \rightarrow {}^{q}C$' is a context variable.

G) $\pi(\forall x\text{-}y\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{p}(y)\circ x], \mathbb{p}[x/\mathbb{p}(y)])$ where x:'$C \rightarrow {}^{q}C$' is a context variable.

H) $\pi(\forall\downarrow(x,z{=}t)\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{c}_C\circ\downarrow(x, \pi(t, \mathbb{c}, \mathbb{p}))], \mathbb{p}[x/\mathbb{c}_C])[z\leftarrow\pi(t, \mathbb{c}, \mathbb{p})]$
where x:'$D,C \rightarrow {}^{q}C$' is a context variable.

I) $\pi(\forall\downarrow(x,z{=}t)\text{-}y\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{p}(y)\circ\downarrow(x, \pi(t, \mathbb{c}, \mathbb{p}))], \mathbb{p}[x/\mathbb{p}(y)])[z\leftarrow\pi(t, \mathbb{c}, \mathbb{p})]$
where x:'$D,C \rightarrow {}^{q}C$' is a context variable.

The variables $y_1,\dots,y_m$ in the rules for the existential quantifier below are the free domain variables in $\mathcal{G}$.

J) $\pi(\exists x\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \pi(\mathcal{G}, \mathbb{c}, \mathbb{p})[x \leftarrow g(s_1,\dots,s_k, y_1,\dots,y_m)]$ where x is a domain variable of sort D,
$s_i := p_i(\downarrow(\mathbb{c}_{C_i}, 0_{C_i}))$ where $p_i{:}C_i \to I_i$ is the i-th element in $\mathcal{SV}(x)$ and
g is a new function symbol of sort $I_1 \times \dots \times I_k \times S(y_1) \times \dots \times S(y_m) \to D$.

K) $\pi(\exists x\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{c}_C \circ x], \mathbb{p}[x/\mathbb{c}_C])[x \leftarrow h(s_1,\dots,s_{j-1},s_{j+1},\dots,s_k,y_1,\dots,y_m)]$

L) $\pi(\exists x\text{-}y\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{p}(y) \circ x], \mathbb{p}[x/\mathbb{p}(y)])[x \leftarrow h(s_1,\dots,s_{j-1},s_{j+1},\dots,s_k,y_1,\dots,y_m)]$
where x:'$C \to {}^{q}C$' is a context variable,
$s_i := p_i(\downarrow(\mathbb{c}_{C_i}, 0_{C_i}))$ where $p_i{:}C_i \to I_i$ is the i-th element in $\mathcal{SV}(x)$, $C_j = C$ and
$h{:}I_1 \times \dots \times I_{j-1} \times I_{j+1} \times \dots \times I_k \times S(y_1) \times \dots \times S(y_m) \to$ '$C \to {}^{q}C$' is a new function symbol.

M) $\pi(\exists\downarrow(x,z{=}t)\ \mathcal{G}, \mathbb{c}, \mathbb{p})$ where x:'$D,C \to {}^{q}C$' is a context variable,
$:= \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{c}_C \circ \downarrow(x, \pi(t, \mathbb{c}, \mathbb{p}))], \mathbb{p}[x/\mathbb{c}_C])[x \leftarrow k(s_1,\dots,s_{j-1},s_{j+1},\dots,s_k,y_1,\dots,y_m), z \leftarrow \pi(t, \mathbb{c}, \mathbb{p})]$

N) $\pi(\exists\downarrow(x,z{=}t)\text{-}y\ \mathcal{G}, \mathbb{c}, \mathbb{p})$ where x:'$D,C \to {}^{q}C$' is a context variable,
$:= \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{p}(y) \circ \downarrow(x, \pi(t, \mathbb{c}, \mathbb{p}))], \mathbb{p}[x/\mathbb{p}(y)])[x \leftarrow k(s_1,\dots,s_{j-1},s_{j+1},\dots,s_k,y_1,\dots,y_m), z \leftarrow \pi(t, \mathbb{c}, \mathbb{p})]$
$s_i := p_i(\downarrow(\mathbb{c}_{C_i}, 0_{C_i}))$ where $p_i{:}C_i \to I_i$ is the i-th element in $\mathcal{SV}(x)$, $C_j = C$ and
$k{:}I_1 \times \dots \times I_{j-1} \times I_{j+1} \times \dots \times I_k \times S(y_1) \times \dots \times S(y_m) \to$ '$D,C \to {}^{q}C$' is a new function symbol.

O) $\pi(\wp\ t\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{c}_C \circ \pi(t, \mathbb{c}, \mathbb{p})], \mathbb{p})$ where t is a term of sort '$C \to {}^{q}C$'.

P) $\pi(\mathcal{G}_1 \wedge \mathcal{G}_2, \mathbb{c}, \mathbb{p}) := \pi(\mathcal{G}_1, \mathbb{c}, \mathbb{p}) \wedge \pi(\mathcal{G}_2, \mathbb{c}, \mathbb{p})$.

Q) $\pi(\mathcal{G}_1 \vee \mathcal{G}_2, \mathbb{c}, \mathbb{p}) := \pi(\mathcal{G}_1, \mathbb{c}, \mathbb{p}) \vee \pi(\mathcal{G}_2, \mathbb{c}, \mathbb{p})$. ■

Notice that the translation rules contain a very strong Skolemization rule for functional context variables. The Skolem functions for a context C do not depend on the universally quantified context variables of the context C itself. In particular when there is only one context, this means they depend only on the domain variables, not on the other context variables. This works because a Skolem function, say g, for a functional context variable whose quantification occurred in the scope of a universal quantifier over another functional context variable, say x, becomes part of a term x∘g that depends on x. If we write this term in second order syntax we see that g(x) really depends on x. The soundness of this strong Skolemization actually depends on a certain condition on the interpretation of the functional context sorts (see. def. 4.3.6 below). The introduction of the stronger Skolemization rule is the reason for including Skolemization into the translation from CL to OSPL.

Proposition 4.3.3 $\Pi_{\mathcal{T}}$ maps well formed CL-formulae to well formed OSPL-formulae.

Proof: A simple check of the translation rules in 4.3.2 and application of def. 3.4.1,4 confirms that the new context terms in the $\mathbb{c}$- and $\mathbb{p}$-argument of π are well sorted. We show the main statement with induction on the size of domain formulae. In most cases it is sufficient to show that terms do not change their sort.

Base Case: This is the variable case which is trivial because variables are not changed.

Induction Step: The induction hypothesis states that translated terms of smaller size have not changed their sorts and translated terms and formulae of smaller size are well formed OSPL-formulae. We perform a case analysis according to the structure of the domain formulae. The interesting cases where the translation rules in 4.3.2 produce new terms are:

Case $f(t_1,\dots,t_n)$ and f is a function symbol.

The translation rule is

B) "$\pi(f(t_1,\dots,t_n), \mathbb{c}, \mathbb{p}) := \Pi_\Sigma(f)(s_1,\dots,s_k, \pi(t_1, \mathbb{c}, \mathbb{p}),\dots, \pi(t_n, \mathbb{c}, \mathbb{p}))$ where f is a function symbol.
The s_i are determined as follows:
If f is a context function then $k = 0$.
Otherwise $s_i := p_i(\downarrow(\mathbb{c}_{C_i}, 0_{C_i}))$ where $p_i{:}C_i \to I_i$ is the i-th element in $\mathcal{SV}(f)$."

According to the induction hypothesis, the sorts of the t_i do not change.
The sort of s_i is the rangesort of the p_i or smaller. Therefore the translated term matches the declaration for $\Pi_\Sigma(f)$ in def. 4.3.1,4, thus keeping its sort.

Case $\pm P(t_1,\ldots,t_n)$ and P is a domain predicate.

The translation rule is

C) "$\pi(\pm P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}) := \pm\Pi_\Sigma(P)(s_1,\ldots,s_k, \pi(t_1, \mathbb{c}, \mathbb{p},\ldots, \pi(t_n, \mathbb{c}, \mathbb{p}))$

The s_i are determined as in the previous case."

The arguments are also the same as in the previous case.

Case $\pm P(t_1,\ldots,t_n)$ and P is a context predicate.

The translation rule is

D) " $\pi(\pm P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}) := \pm P(s_1,\ldots,s_n)$

The s_i are determined as follows:

If t_i:'$C\rightarrow{}^qC$' is either a context variable x or a term $\downarrow(x,z)$ then $s_i := \downarrow((\mathbb{p}(x)\circ t_i)), 0_C)$

If $t_i = s \circ t$ and s is a variable x:'$C\rightarrow{}^qC$' or a term $\downarrow(x,z)$ then $s_i := \downarrow((\mathbb{p}(x)\circ s\circ\pi(t, \mathbb{c}, \mathbb{p})), 0_C)$

otherwise $s_i := \pi(t_i, \mathbb{c}, \mathbb{p})$"

The sort of s_i is C or smaller in all three cases.Since $P(t_1,\ldots,t_n)$ is a well formed domain atom, the sort of the t_i-position must be C or larger (def. 4.1.3,ii). Thus, the translated atom is a well sorted OSPL-atom.

Case $\forall\downarrow(x,z=t)\ \mathcal{G}$ where x:'$D,C\rightarrow{}^qC$' is a context variable.

The translation rule is

H) "$\pi(\forall\downarrow(x,z=t)\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}[\mathbb{c}_C/\mathbb{c}_C\circ\downarrow(x, \pi(t, \mathbb{c}, \mathbb{p}))], \mathbb{p}[x/\mathbb{c}_C])[z\leftarrow\pi(t, \mathbb{c}, \mathbb{p})]$"

By definition, z and t have the same sorts (def.4.1.3,iv). Exploiting the induction hypothesis for $\pi(t, \mathbb{c}, \mathbb{p})$ we can safely replace z by the translated term without changing the sorts of terms.

The same holds for the case $\forall\downarrow(x,z=t)\text{-}y\ \mathcal{G}$.

Case $\exists x\ \mathcal{G}$ where x is a domain variable of sort D.

The translation rule is

J) "$\pi(\exists x\ \mathcal{G}, \mathbb{c}, \mathbb{p}) := \pi(\mathcal{G}, \mathbb{c}, \mathbb{p})[x\leftarrow g(s_1,\ldots,s_k, y_1,\ldots,y_m)]$

$s_i := p_i(\downarrow(\mathbb{c}_{Ci}, 0_{Ci}))$ where $p_i:C_i\rightarrow I_i$ is the i-th projector function in $\mathcal{SU}(x)$,

and g is a new function symbol of sort $I_1\times\ldots\times I_k\times S(y_1)\times\ldots\times S(y_m) \rightarrow D$."

$g(s_1,\ldots,s_k, y_1,\ldots,y_m)$ is a well sorted term and its sort is D. Therefore x of sort D is replaced by a term of sort D which does not change the sorts and this holds for the remaining rules with existential quantifier as well. ■

Examples for the Translation of Domain Formulae

We assume C is the only basic context sort and therefore it is also the interpretation sort.

Let the symbol variation function map all symbols to the identity function. We shall omit it.

$\Pi_{\mathcal{J}}(P)$

$= \quad P'(\downarrow(\text{identity}, 0_C)) \qquad (\rightarrow P'(0_C))$

$\Pi_{\mathcal{J}}(\forall x\text{:'}C\rightarrow C\text{' } P)$

$= \quad \forall x\text{:'}C\rightarrow C\text{' } P'(\downarrow(x, 0_C))$

$\Pi_{\mathcal{J}}(\forall x\text{:'}C\rightarrow C\text{' } a = b)$ (We do not distinguish between the object and meta equality symbol)

$= \quad \forall x\text{:'}C\rightarrow C\text{' } a(\downarrow(x, 0_C)) = b(\downarrow(x, 0_C))$

$\Pi_{\mathcal{J}}(\forall x{:}D\ \forall\downarrow(y\text{:'}D,C\rightarrow C\text{'}, z=f(x))\ Q(x))$

$= \quad \forall x{:}D\ \forall y\text{:'}D,C\rightarrow C\text{' } Q'(\downarrow(\downarrow(y, f(x)), 0_C), x)$

$\Pi_{\mathcal{J}}(\forall x{:}A\ \forall y\text{:'}C\rightarrow C\text{' } \exists z{:}D\ R(x, z, a))$

$= \quad \forall x{:}A\ \forall y\text{:'}C\rightarrow C\text{' } R'(\downarrow(y, 0_C), x, g_z(\downarrow(y, 0_C), x), a(\downarrow(y, 0_C)))$

$\Pi_{\mathcal{J}}(\forall x{:}A\ \forall y\text{:'}C\rightarrow C\text{' } \exists z\text{:'}C\rightarrow C\text{' } Q(x))$

$= \quad \forall x{:}A\ \forall y\text{:'}C\rightarrow C\text{' } Q'(\downarrow(y\circ h_z(x), 0_C)\ x)$

h needs not depend explicitly on y because the whole term $y\circ h(x)$ depends on y.

Let S be a context predicate of sort $C\times C\times D$

$\Pi_{\mathcal{J}}(\forall x{:}D\ \exists y\text{:'}C\rightarrow C\text{' } \forall z\text{:'}C\rightarrow C\text{' } \exists u\text{:'}C\rightarrow C\text{' } \exists w\text{:'}C\rightarrow C\text{'-}u\ \ S(z\circ u, z\circ w, a) \vee \forall p\text{:'}C\rightarrow C\text{' } Q(x))$

$= \quad \forall x{:}D\ \forall z\text{:'}C\rightarrow C\text{'}\forall v\text{:'}C\rightarrow C\text{' } S(\downarrow(h_y(x)\circ z\circ h_u(x), 0_C), \downarrow(h_y(x)\circ z\circ h_w(x), 0_C), a(\downarrow(h_y(x)\circ z\circ h_w(x), 0_C))) \vee$

$\forall p\text{:'}C\rightarrow C\text{' } Q'((\downarrow(h_y(x)\circ z\circ h_w(x)\circ p, 0_C))), x)$

$\Pi_{\mathcal{F}}(\forall x{:}`C{\to}C'\ \wp\ f(x)\ P) = \forall x{:}`C{\to}C'\ P'(\downarrow(x \circ f(x), 0_C), x)$
where $f{:}`C{\to}C' \to `C{\to}C'$ is a context function.

Assume B and C are the basic context sorts and $\mathcal{SV}$ maps to the identity function.
$\Pi_{\mathcal{F}}(\forall x{:}`B{\to}B\ \forall y{:}`C{\to}C\ P)$
$= \forall x{:}`B{\to}B\ \forall y{:}`C{\to}C\ P''(\downarrow(x, 0_B), \downarrow(y, 0_C))$

$\Pi_{\mathcal{F}}(\forall x{:}`B{\to}B\ \forall y{:}`C{\to}C\ \exists u{:}`B{\to}B\ \exists v{:}`C{\to}C\ P)$
$= \forall x{:}`B{\to}B\ \forall y{:}`C{\to}C\ P''(\downarrow(x \circ f_u(\downarrow(y, 0_C)), 0_B), \downarrow(y \circ f_v(\downarrow(x, 0_B)), 0_C))$
where $f_u{:}C \to `B{\to}B'$ and $f_v{:}B \to `C{\to}C'$ are Skolem functions.

For the next example, assume we have two basic context sorts, W, and WP and the single interpretation context sort is W. The symbol variation function maps all symbols to a function PW:WP→W.
$\Pi_{\mathcal{F}}(P)$
$= P'(PW(\downarrow(\text{identity}, 0_{WP})))\quad (\to P'(PW(0_{WP})) \to P'(0_W)$ if PW is the projector function)
$\Pi_{\mathcal{F}}(\forall x{:}WP{\to}WP\ Q(a))$
$= \forall x{:}WP{\to}WP\ Q'(PW(\downarrow(x, 0_{WP})), a(PW(\downarrow(x, 0_{WP}))))$ ■

The **specification morphism** $\Pi_{\mathcal{S}}$ needs only apply the signature morphism to the signature and the formula morphism to the formulae. Apart from the special constants 0_C and the Skolem functions, no additional symbols and axioms are generated.

The next thing to do is to define the interpretation morphism $\Pi_{\mathfrak{I}}$ together with its inverse $\Pi_{\mathfrak{I}}^{-1}$. The proof that the interpretation morphism is well defined, i.e. the proof that it really maps CL-models for CL-specifications to OSPL-models for the translated specifications and vice versa is at the same time the soundness and completeness proof for the translation itself.

$\Pi_{\mathfrak{I}}$ maps all the different Σ-structures (worlds) of the CL-interpretation into one big Σ-structure where the former contexts (worlds) become part of the domain and the additional arguments of the translated function and predicate symbols are used to represent the dependence of the function values from the contexts. The information about the possible worlds structure, i.e. the accessibility relation is still there in form of the functional interpretation of the functional context sorts. The inverse of $\Pi_{\mathfrak{I}}$ decomposes this big Σ-structure into its constituents.

Definition 4.3.4 The Interpretation Morphism $\Pi_{\mathfrak{I}}$
The interpretation morphism $\Pi_{\mathfrak{I}}$ has to translate CL-interpretations into OSPL-interpretations.
Given a CL-interpretation $\mathfrak{I} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ over a CL-signature Σ consisting of the domain part Σ_D and the context part Σ_C and with basic context sorts $C_1,\ldots,C_m$ and interpretation context sorts $(I_1,\ldots,I_k)$, we construct a functional $\Pi_\Sigma(\Sigma)$-structure $\mathcal{M}$ as follows:

1. The domain of $\mathcal{M}$:
   For the basic context sorts C: $C_{\mathcal{M}} := C_{\mathcal{C}}$.
   For the domain sorts D: $D_{\mathcal{M}} := D_{\mathcal{SV}}$.
   The interpretation of the functional sorts in $\mathcal{M}$ is as in theorem 3.4.3.
2. The functions in $\mathcal{M}$:
   For the context functions: $f_{\mathcal{M}} := f_{\mathcal{C}}$
   Let f be an n-place domain function symbol in Σ.

We define $f_{\mathcal{M}}$ as an k+n-place function such that the following axioms hold:

$\forall i_1 \in I_{1C}, \ldots, i_k \in I_{kC}, a_1, \ldots, a_n$

If $f_{\mathcal{SV}(i1,\ldots,ik)}(a_1,\ldots,a_n)$ is defined (depending on the sorts of f)

then $f_{\mathcal{M}}(i_1,\ldots,i_k,a_1,\ldots,a_n) = f_{\mathcal{SV}(i1,\ldots,ik)}(a_1,\ldots,a_n)$.

For the special constant symbols 0_C we set $0_{C\mathcal{M}} := \mathcal{C}_C$ where $\mathcal{C}_C$ is the element in $\mathcal{C}$ that corresponds to the sort C. The interpretation of the additional Skolem function symbols of $\Pi_\Sigma(\Sigma)$ in $\mathcal{M}$ depends on the actual specification to be translated and is therefore left open for the moment.

3. The relations in $\mathcal{M}$:

The equation symbol is again interpreted as the identity.

For the context predicates: $P_{\mathcal{M}} := P_{\mathcal{C}}$

Let P be an n-place domain predicate symbol in Σ.

We define $P_{\mathcal{M}}$ as an k+n-place relation such that the following axiom holds:

$\forall i_1 \in I_{1C}, \ldots, i_k \in I_{kC}, a_1, \ldots, a_n$

If $P_{\mathcal{SV}(i1,\ldots,ik)}(a_1,\ldots,a_n)$ is defined then $P_{\mathcal{M}}(i_1,\ldots,i_k,a_1,\ldots,a_n)$ iff $P_{\mathcal{SV}(i1,\ldots,ik)}(a_1,\ldots,a_n)$.

It is now easy to verify that $\Pi_{\mathfrak{I}}(\mathfrak{I}) := (\mathcal{M}, \mathcal{V})$ is really a $\Pi_\Sigma(\Sigma)$-interpretation.

4. The inverse $\Pi_{\mathfrak{I}}^{-1}$ of the interpretation morphism is defined as follows:

Let $\mathcal{M}$ be a functional $\Pi_\Sigma(\Sigma)$-structure. First of all we construct a CL-frame $(\mathcal{C}, \mathcal{SV}) = \Pi_P^{-1}(\mathcal{M})$:

a) $\mathcal{C}$ is the substructure of $\mathcal{M}$ related to the context sorts and context symbols.

b) $\mathcal{SV}$ assigns to each tuple $(i_{1C},\ldots,i_{kC}) \in I_{1C} \times \ldots \times I_{kC}$ the following Σ_D-structure $\mathcal{A} = \mathcal{A}_{i1,\ldots,ik}$:

➤ The domain of $\mathcal{A}$: For every domain sort D: $D_{\mathcal{A}} := \mathcal{D}_{\mathcal{M}}$.

➤ The functions in $\mathcal{A}$:

Let f be an k+n-place domain function symbol in $\Pi_\Sigma(\Sigma)$.

We define $f_{\mathcal{A}}$ as an n-place function such that the following axioms hold:

$\forall i_1 \in I_{1C}, \ldots, i_k \in I_{kC}, a_1, \ldots, a_n$

If $f_{\mathcal{M}}(i_1,\ldots,i_k, a_1,\ldots,a_n)$ is defined then $f_{\mathcal{A}}(a_1,\ldots,a_n) = f_{\mathcal{M}}(i_1,\ldots,i_k, a_1,\ldots,a_n)$.

➤ The relations in $\mathcal{A}$:

Let P be an k+n-place domain predicate symbol in $\Pi_\Sigma(\Sigma)$.

We define $P_{\mathcal{A}}$ as an n-place relation such that the following axiom holds:

$\forall i_1 \in I_{1C}, \ldots, i_k \in I_{kC}, a_1, \ldots, a_n$

If $P_{\mathcal{A}}(a_1,\ldots,a_n)$ is defined then $P_{\mathcal{A}}(a_1,\ldots,a_n)$ iff $P_{\mathcal{M}}(i_1,\ldots,i_k,a_1,\ldots,a_n)$.

For an OSPL-interpretation $\mathfrak{I} = (\mathcal{M}, \mathcal{V})$ we define $\Pi_{\mathfrak{I}}\text{-}1(\mathfrak{I}) := (\Pi_{\mathfrak{I}}\text{-}1(\mathcal{M}), \mathcal{V}, (0_{C1\mathcal{M}},\ldots, 0_{Cm\mathcal{M}}), \emptyset)$ (the context assignment $\mathcal{P}$ is left open.) ■

Soundness of the Translation

In this section we show that the logic morphism Π translates a CL-specification that is satisfied by a CL-interpretation $\mathfrak{I}$ into an OSPL-specification that is satisfied by the OSPL-interpretation $\Pi_{\mathfrak{I}}(\mathfrak{I})$. Since the context part of the specification is already an OSPL-specification we need to consider only the domain part.

We have used a very strong Skolemization of the existentially quantified context access variables. Their Skolem functions do not depend on their "own context". The soundness of this strong Skolemization depends on the fact that the set of context access functions assigned to a functional context sort 'C→C' is sufficiently rich. To illustrate what this means, suppose we have a CL-formula $\forall$x:'C→C' $\exists$y:'C→C' P which is translated into $\forall$x:'C→C' P$\downarrow$(x$\circ$y, 0_C) where y is no longer a variable, but the Skolem constant for the original y. The usual Skolemization rule would generate the formula $\forall$x:'C→C' P$\downarrow$(x$\circ$y(x), 0_C) where y depends on x. The interpretation of the term $\downarrow$(x$\circ$y(x), 0_C) for the various $x_i \in$ 'C→C'$_{\mathcal{C}}$ is:

$$0_{C\mathcal{C}} \xrightarrow{x_1} c_1 \xrightarrow{y_1} c_1'$$

$$\ldots$$

$$0_{C\mathcal{C}} \xrightarrow{x_k} c_k \xrightarrow{y_k} c_k'$$

$$\ldots$$

and all these y_i are in 'C→C'$_{\mathcal{C}}$. According to the semantics of the existential quantifier we can w.l.o.g assume that if $x_i(0_{C\mathcal{C}}) = x_j(0_{C\mathcal{C}})$ for some x_i and x_j then $y_i(x_i(0_{C\mathcal{C}})) = y_j(x_j(0_{C\mathcal{C}}))$. If we can also assume that 'C→C'$_{\mathcal{C}}$ is sufficiently rich to contain an additional y such that $y_i(c_i) = y(c_i)$ for all these y_i and c_i, we can use this y instead of all the y_i, and we have made the Skolem function y independent of x. However, we need not require for all sets $\{(c_i, y_i(c_i)) \mid c_i \in C_{\mathcal{C}}, y_i \in$ C→C'$_{\mathcal{C}}\}$ of such tuples the existence of an appropriate y, but only for the sets of tuples which really occur during the interpretation of a formula. The restrictions are: The c_i are obtained by the application of some $x_i \in$ C→C'$_{\mathcal{C}}$ to $0_{C\mathcal{C}}$ and the y_i are functions which are considered as interpretations of the $\exists$-quantified variables. In chapter 5 we shall see an example where the $\exists$-quantification is restricted to require for two different x which are in a certain sense correlated the existence of a unique y. This restriction can be exploited to require the existence of y only for certain cases, i.e. to impose conditions on the interpretations of the functional context sorts.

To ensure that the context access functions are rich enough to allow the strong Skolemization we require for every interpretation of a context specification that the following "$\exists$-quantifier independency lemma" holds:

Definition 4.3.5 The $\exists$-Quantifier Independency Lemma

Let $\mathfrak{I} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ be a CL-interpretation over a CL-signature Σ, where 'C →^{q}C' and 'D,C →^{q}C' are functional context sorts.

➤ Let $\mathcal{F}$ be a domain formula over Σ, $\mathcal{F}$ containing an existential quantifier over a context access variable x. The set of all 'C →^{q}C'$_{\mathcal{C}}$ or 'D,C →^{q}C'$_{\mathcal{C}}$ functions respectively which are used as interpretations of this particular occurrence of x is called an *$\exists$-quantified set* of context access functions (for $\mathcal{F}$). (This set is determined essentially by the embracing universally quantified variables. In certain cases there are additional restrictions.)

➤ The *$\exists$-quantifier independency lemma* requires the following two statements to hold:

a) For all $c \in C_{\mathcal{C}}$ and for all $\exists$-quantified sets $\mathcal{Y} \subseteq$ 'C →^{q}C'$_{\mathcal{C}}$ there exists an $y \in \mathcal{Y}$ such that
$\forall x_i \in$ 'C→C'$_{\mathcal{C}}$, $y_i \in \mathcal{Y}$: $y(x_i(c)) = y_i(x_i(c))$ where 'C→C' is the top sort in the 'C →^{q}C'-sorts.

b) For all $c \in C_{\mathcal{C}}$, $l \in D_{\mathcal{C}}$ and for all $\exists$-quantified sets $\mathcal{Y} \subseteq$ 'D,C →^{q}C'$_{\mathcal{C}}$ there exists an $y \in \mathcal{Y}$ such that
$\forall x_i \in$ 'C→C'$_{\mathcal{C}}$, $y_i \in \mathcal{Y}$: $y(l, x_i(c)) = y_i(l, x_i(c))$ where 'C→C' is again the top sort in the 'C →^{q}C'-sorts.■

This lemma has to be proved for every source logic that is translated via CL into OSPL. When there are no restrictions on the interpretations of the functional context sorts, this is usually no problem. In case the context access functions are restricted by axioms in the context specification, there must be a corresponding restriction on the interpretation of the existentially quantified variables.

In the sequel we always assume the ∃-Quantifier Independency Lemma to hold.

The satisfiability relation for OSPL (def. 3.3.3) is defined inductively over the structure of OSPL-formulae. Satisfiability has therefore to be proved by structural induction on OSPL-formulae. The only thing we know, however, is that $\mathfrak{I}$ is a model for the original CL-formula $\mathcal{F}$, and this is a property that holds for $\mathcal{F}$ itself. It is therefore not an adequate basis for an essentially "bottom up" structural induction. The proof idea is therefore to follow the "top down" recursion of π and to decompose $\mathfrak{I}$ into the interpretations for the subformulae $\mathcal{G}$ in $\mathcal{F}$. When we have reached the atomic level we can translate the CL-interpretations for the CL-atoms into OSPL-interpretations for the translated atoms and from these build up the OSPL-model for the translated formula. To this end we augment the formula morphism $\Pi_{\mathcal{F}}$ (def. 4.3.2) with an additional argument that takes a CL-model for the CL-formula to be translated.

Definition 4.3.6 The Augmented Formula Morphism $\Pi_{\mathcal{F}}$

The augmented formula morphism $\Pi_{\mathcal{F}}$ takes a CL-formula $\mathcal{F}$ (we consider only context formulae) and a CL-model for $\mathcal{F}$. The CL-model is decomposed into the CL-models for the corresponding subformulae of $\mathcal{F}$. As a side effect we fix the yet undefined semantics of the generated Skolem functions. The augmented auxiliary function π takes as an additional argument the list $\mathfrak{I}$ of CL-interpretations for the currently being translated subformula of $\mathcal{F}$.

The toplevel call for π is:

$\Pi(\mathcal{F}, \mathfrak{I}) = \pi(\mathcal{F}, \mathbb{c}_0, \mathbb{p}_0, \{\mathfrak{I}\})$ where $\mathbb{c}_0$ and $\mathbb{p}_0$ are the default values of def. 4.3.2.

Notational Conventions:

We abbreviate $\Pi_{\mathfrak{I}}(\mathfrak{I})$ with $\mathfrak{I}_{PL}$.

In order to distinguish between the satisfiability relation for domain formulae (def. 4.2.5) and the OSPL-satisfiability relation (def. 3.3.3) we write the former as $\models_C$ and the latter as $\models_P$.

(Notice in the sequel the similarities between the $\mathbb{c}$ and $\mathbb{p}$ parameters of π on the syntactical side and the $[\ldots]_{\mathcal{C}}$ and $[\ldots]_{\mathcal{P}}$ components of $\mathfrak{I}$ on the semantical side. This is no coincidence.)

$\pi(\mathcal{F}, \mathbb{c}, \mathbb{p}, \mathfrak{I})$ is again defined inductively over the structure of $\mathcal{F}$:

A) $\pi(x, \mathbb{c}, \mathbb{p}, \mathfrak{I}) := x$

B) $\pi(f(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}, \mathfrak{I}) := \Pi_\Sigma(f)(s_1,\ldots,s_k, \pi(t_1, \mathbb{c}, \mathbb{p}, \mathfrak{I}),\ldots, \pi(t_n, \mathbb{c}, \mathbb{p}, \mathfrak{I}))$

C) $\pi(\pm P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}, \mathfrak{I}) := \pm\Pi_\Sigma(P)(s_1,\ldots,s_k, \pi(t_1, \mathbb{c}, \mathbb{p}, \mathfrak{I}),\ldots, \pi(t_n, \mathbb{c}, \mathbb{p}, \mathfrak{I}))$

D) $\pi(\pm P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}, \mathfrak{I}) := \pm P(s_1,\ldots,s_n)$ where P is a context predicate.

The s_i are determined as follows:

If t_i:'$C \rightarrow {}^qC$' is a context variable x or a term $\downarrow(x,z)$ then $s_i := \downarrow((\mathbb{p}(x) \circ t_i)), 0_C)$.

If $t_i = s \circ t$ and s is a variable x:'$C \rightarrow {}^qC$' or a term $\downarrow(x,z)$ then $s_i := \downarrow((\mathbb{p}(x) \circ s \circ \pi(t, \mathbb{c}, \mathbb{p}, \mathfrak{I})), 0_C)$,

otherwise $s_i := \pi(t_i, \mathbb{c}, \mathbb{p}, \mathfrak{I})$

E) $\pi(\forall x\ \mathcal{G}, \mathbb{c}, \mathbb{p}, \mathfrak{I}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}, \mathbb{p}, \mathfrak{I}')$ where x:D is a domain variable.

$\mathfrak{I}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \mathfrak{I}, \chi \in D_{\mathcal{SV}}\}$

F) $\pi(\forall x\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/c_C \circ x], \mathbb{p}[x/c_C], \boldsymbol{\mathfrak{I}}')$ where x:'C→^{q}C' is a context variable.

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}\}$

G) $\pi(\forall x\text{-}y\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/\mathbb{p}(y)\circ x], \mathbb{p}[x/\mathbb{p}(y)], \boldsymbol{\mathfrak{I}}')$ where x:'C→^{q}C' is a context variable.

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}\}$

H) $\pi(\forall{\downarrow}(x,z{=}t)\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/c_C \circ {\downarrow}(x, \pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}))], \mathbb{p}[x/c_C], \boldsymbol{\mathfrak{I}}')[z \leftarrow \pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})]$

where x:'D,C→^{q}C' is a context variable.

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}\}$

I) $\pi(\forall{\downarrow}(x,z{=}t)\text{-}y\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/\mathbb{p}(y) \circ {\downarrow}(x, \pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}))], \mathbb{p}[x/\mathbb{p}(y)], \boldsymbol{\mathfrak{I}}')[z \leftarrow \pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})]$

where x:'D,C→^{q}C' is a context variable.

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}\}$

J) $\pi(\exists x\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}')[x \leftarrow g(s_1,\ldots,s_k, y_1,\ldots,y_l)]$ where x:D is a domain variable.

Among the possible interpretations for g we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in D_{\mathcal{SV}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}} \models_C \mathcal{G}$

there is an χ' with $\mathfrak{I}_{PL}(g(s_1,\ldots,s_k, y_1,\ldots,y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in D_{\mathcal{SV}}, \mathfrak{I}' \models_C \mathcal{G}\}$

K) $\pi(\exists x\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c[C/c_C \circ x], \mathbb{p}[x/c_C], \boldsymbol{\mathfrak{I}}')[x \leftarrow h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)]$

where x:'C→^{q}C' is a context variable.

Among the possible interpretations for h we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \models_C \mathcal{G}$

there is an χ' with $\mathfrak{I}_{PL}(h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}\ \mathfrak{I}' \models_C \mathcal{G}\}$

L) $\pi(\exists x\text{-}y\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c[C/\mathbb{p}(y)\circ x], \mathbb{p}[x/\mathbb{p}(y)], \boldsymbol{\mathfrak{I}}')[x \leftarrow h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)]$

where x:'C→^{q}C' is a context variable.

Among the possible interpretations for h we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \models_C \mathcal{G}$

there is an χ' with $\mathfrak{I}_{PL}(h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}\ \mathfrak{I}' \models_C \mathcal{G}\}$

M) $\pi(\exists{\downarrow}(x,z{=}t)\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})$ where x:'D,C→^{q}C' is a context variable.

$:= \pi(\mathcal{G}, c[C/c_C \circ {\downarrow}(x,\pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}))], \mathbb{p}[x/c_C], \boldsymbol{\mathfrak{I}}')[x \leftarrow k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m), z \leftarrow \pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})]$

Among the possible interpretations for k we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[x/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \models_C \mathcal{G}$

there is an χ' with $\mathfrak{I}_{PL}(k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}D,C\rightarrow{}^qC\text{'}_{\mathcal{C}}\ \mathfrak{I}' \models_C \mathcal{G}\}$

N) $\pi(\exists{\downarrow}(x,z{=}t)\text{-}y\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})$ where x:'D,C→^{q}C' is a context variable.

$:= \pi(\mathcal{G}, c[C/\mathbb{p}(y) \circ {\downarrow}(x,\pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}))], \mathbb{p}[x/\mathbb{p}(y)], \boldsymbol{\mathfrak{I}}')[x \leftarrow k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m), z \leftarrow \pi(t,c,\mathbb{p},\boldsymbol{\mathfrak{I}})]$

Among the possible interpretations for k we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in \text{`}C\rightarrow{}^qC\text{'}_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[x/\chi(\mathfrak{I}(t), \mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \models_C \mathcal{G}$

there is a χ' with $\mathfrak{I}_{PL}(k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{`}D,C\rightarrow{}^qC\text{'}_{\mathcal{C}}\ \mathfrak{I}' \models_C \mathcal{G}\}$

O) $\pi(\wp\ t\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c[C/c_C \circ \pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})], \mathbb{p}, \boldsymbol{\mathfrak{I}}')$ where t:'C→^{q}C' is a context access term.

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[C/\mathfrak{I}(t)(\mathcal{C}_C)]_{\mathcal{C}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$

P) $\pi(\mathcal{G}_1 \wedge \mathcal{G}_2, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}_1, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) \wedge \pi(\mathcal{G}_2, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})$

Q) $\pi(\mathcal{G}_1 \vee \mathcal{G}_2, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}_1, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}_1) \vee \pi(\mathcal{G}_2, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}_2)$

$\boldsymbol{\mathfrak{I}}_i := \{\mathfrak{I} \in \boldsymbol{\mathfrak{I}} \mid \mathfrak{I} \models_C \mathcal{G}_i\}\ i = 1,2$ ■

Before we can use this definition we must prove that it is well defined, i.e. that the CL-interpretations which are decomposed from a CL-model really satisfy the corresponding subformulae and that the Skolem functions are interpreted as total functions.

Lemma 4.3.7 The Augmented Formula Morphism is Well Defined.

a) If $\mathfrak{I}_0$ is a CL-model for a CL-specification $\mathcal{S}$ then for each recursive call $\pi(\mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})$ during the translation of the formulae in $\mathcal{S}$, the following invariant holds as long as $\mathcal{G}$ is a formula: For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: $\mathfrak{I} \models_C \mathcal{G}$.

b) The Skolem functions are interpreted as total functions.

Proof of a (b is shown inside this proof): By induction on the recursion depth.

Base Case: Recursion depth = 0: This is just the precondition saying that $\mathfrak{I}_0$ satisfies $\mathcal{S}$.

Induction Step: Let the recursion depth be greater than 0.

Let $\pi(\mathcal{F}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})$ be the actual call to π. The induction hypothesis states: For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: $\mathfrak{I} \models_C \mathcal{F}$.

In order to show that the statement also holds for the next recursion step we must perform a case analysis according to the structure of $\mathcal{F}$ and analyze the corresponding translation rule.

Case $\mathcal{F} = \forall x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

E) $\pi(\forall x\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}')$ $\qquad \boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in D_{\mathcal{SV}}\}$

The induction hypothesis immediately implies $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}'$: $\mathfrak{I} \models_C \mathcal{G}$. (def. 4.2.5)

The same holds for the remaining three cases with universal quantifiers.

Case $\mathcal{F} = \exists x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

J) $\pi(\exists x\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}')[x \leftarrow g(s_1, \ldots, s_k, y_1, \ldots, y_m)]$

Among the possible interpretations for g we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in D_{\mathcal{SV}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}} \models_C \mathcal{G}$

there is an χ' with $\mathfrak{I}_{PL}(g(s_1, \ldots, s_k, y_1, \ldots, y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in D_{\mathcal{SV}}, \mathfrak{I}' \models_C \mathcal{G}\}$.

With "$\mathfrak{I}' \models_C \mathcal{G}$" the condition a) is explicitly enforced.

Since $\mathfrak{I} \models_C \exists x\ \mathcal{G}$, there is at least one $\chi \in D_{\mathcal{SV}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}} \models_C \mathcal{G}$.

Therefore the semantics of the Skolem function g is well defined (statement b).

Case $\mathcal{F} = \exists x\ \mathcal{G}$ and x:'C$\rightarrow^q$C' is a context variable.

The translation rule is:

K) $\pi(\exists x\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c[C/c_C \circ x], \mathbb{p}[x/c_C], \boldsymbol{\mathfrak{I}}')[x \leftarrow h(s_1, \ldots, s_{j-1}, s_{j+1}, \ldots, s_k, y_1, \ldots, y_m)]$

Among the possible interpretations for h we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in$ 'C$\rightarrow^q$C'$_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \models_C \mathcal{G}$

there is a χ' with $\mathfrak{I}_{PL}(h(s_1, \ldots, s_{j-1}, s_{j+1}, \ldots, s_k, y_1, \ldots, y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in$ 'C$\rightarrow^q$C'$_{\mathcal{C}}$ $\mathfrak{I}' \models_C \mathcal{G}\}$.

Again with "$\mathfrak{I}' \models_C \mathcal{G}$" the condition a) is explicitly enforced.

The Skolem function h does not depend on the free context variables for the context C. Therefore a function in 'C$\rightarrow^q$C'$_{\mathcal{C}}$ must be available that can serve as interpretation of $h(s_1, \ldots, s_{j-1}, s_{j+1}, \ldots, s_k, y_1, \ldots, y_m)$ under the different assignments of values to these variables. Since we assume the $\exists$-quantifier independency lemma (def. 4.3.5) to hold, C$\rightarrow^q$C'$_{\mathcal{C}}$ is sufficiently rich to contain this function. Therefore the semantics of h is well defined.

The same holds for the remaining cases with existential quantifiers.

Case $\mathcal{F} = \wp\ t\ \mathcal{G}$ and t:'C$\rightarrow^q$C' is a context access term.

The translation rule is

O) $\pi(\wp\ t\ \mathcal{G}, c, \mathbb{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c[C/c_C \circ \pi(t, c, \mathbb{p}, \boldsymbol{\mathfrak{I}})], \mathbb{p}, \boldsymbol{\mathfrak{I}}')$ $\quad \boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[C/\mathfrak{I}(t)(\mathcal{C}_C)]_{\mathcal{C}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$.

Since, according to the induction hypothesis, $\mathfrak{I} \models_C \wp\ t\ \mathcal{G}$, we can apply the semantics definition of $\wp$ (def. 4.2.5) and obtain immediately $\mathfrak{I}[C/\mathfrak{I}(t)(\mathcal{C}_C) \models_C \mathcal{G}$.

The cases with $\wedge$ and $\vee$ are trivial. ■

The next lemma ensures that the OSPL-interpretation of translated terms is identical with the CL-interpretation of the original terms (lemma 4.3.8). The main reason for that is that the $\mathbb{c}$- and $\mathbb{p}$-argument of π, which record the syntactical context, in some sense run in parallel with their semantic counterparts, the $\mathcal{C}$- and $\mathcal{P}$-components of the interpretations in the $\mathfrak{S}$-argument. The correlations between $\mathbb{c}$ and $\mathbb{C}$ as well as between $\mathbb{p}$ and $\mathcal{P}$ are shown in lemma 4.3.9. Lemma 4.3.8 has to be shown first because the interpretation of the translated term t in the $\forall\downarrow(x,z=t)\ldots$ and $\exists\downarrow(x,z=t)\ldots$ cases are needed in the proof of lemma 4.3.9. Its quite complex precondition is shown as a general invariant of π in lemma 4.3.9. In the soundness lemma, 4.3.8 is therefore always applicable.

Lemma 4.3.8 The Interpretation of Terms is Invariant.

Let t be a domain term over a signature Σ and let $\Im_0 = (\mathfrak{F}, \mathcal{V}_0, \mathcal{C}_0, \mathcal{P}_0)$ be a CL-interpretation for Σ.
Let $\pi(t, \mathbb{c}, \mathbb{p})$ be a call to the translation function.
Furthermore let $\Im = (\mathfrak{F}, \mathcal{V}, \mathcal{C}, \mathcal{P})$ be a CL-interpretation for Σ such that for every basic context sort C:

$\Im_{PL}(\mathbb{c}_C)(\mathcal{C}_{0C}) = \mathcal{C}_C$ (✱)

then $\Im(t) = \Im_{PL}(\pi(t, \mathbb{c}, \mathbb{p}))$ where $\Im_{PL} := \Pi_{\Im}(\Im) =: (\mathcal{M}, \mathcal{V})$

Proof: By induction on the size of t.

Base Case: t is a variable.

The statement is true because the translation neither changes variables nor their interpretation.

Induction Step: $t = f(t_1,\ldots,t_n)$

$\Im_{PL}(\pi(t, \mathbb{c}, \mathbb{p})) = \Im_{PL}(\Pi_\Sigma(f)(s_1,\ldots,s_k, \pi(t_1, \mathbb{c}, \mathbb{p}),\ldots, \pi(t_n, \mathbb{c}, \mathbb{p})))$

$s_i := p_i(\downarrow(\mathbb{c}_{Ci}, 0_{Ci}))$ where $p_i{:}C_i \rightarrow I_i$ is the i-th element in $\mathcal{SV}(f)$. (def. 4.3.2,B)

$= f_{\mathcal{M}}(\Im_{PL}(s_1),\ldots,\Im_{PL}(s_k), \Im_{PL}(\pi(t_1, \mathbb{c}, \mathbb{p})),\ldots, \Im_{PL}(\pi(t_n, \mathbb{c}, \mathbb{p}))))$

$= f_{\mathcal{M}}(\Im_{PL}(s_1),\ldots,\Im_{PL}(s_k), \Im(t_1),\ldots, \Im(t_n)))$ (induction hypothesis)

$= f_{\mathcal{M}}(\Im_{PL}(p_1)(\mathcal{C}_{C1}),\ldots,\Im_{PL}(p_k)(\mathcal{C}_{Ck}), \Im(t_1),\ldots, \Im(t_n)))$ (condition ✱)

$= f_{\mathcal{A}}(\Im(t_1),\ldots, \Im(t_n)))$ $\mathcal{A} := \mathcal{SV}(\Im_{PL}(p_1)(\mathcal{C}_{C1}),\ldots,\Im_{PL}(p_k)(\mathcal{C}_{Ck}))$ (def 4.3.4,2)

$\mathcal{A}$ is the actual Σ-structure for f (def. 4.2.3)

$= \Im(t)$ (def. 4.2.4) ■

Lemma 4.3.9 The Correlation Between $\mathbb{c}$ and $\mathcal{C}$, and $\mathbb{p}$ and $\mathcal{P}$.

If $\Im_0 = (\mathfrak{F}, \mathcal{V}_0, \mathcal{C}_0, \mathcal{P}_0)$ is a CL-interpretation satisfying a CL-specification $\mathcal{S}$ then for each recursive call $\pi(\mathcal{G}, \mathbb{c}, \mathbb{p}, \mathfrak{S})$ during the translation of the formulae in $\mathcal{S}$, the following invariants hold:

For every $\Im = (\mathfrak{F}, \mathcal{V}, \mathcal{C}, \mathcal{P}) \in \mathfrak{S}$:

a) for every basic context sort C: $\Im_{PL}(\mathbb{c}_C)(\mathcal{C}_{0C}) = \mathcal{C}_C$ and

b) for every free context variable x in $\mathcal{G}$: $\Im_{PL}(\mathbb{p}(x))(\mathcal{C}_{0C}) = \mathcal{P}(x)$.

Proof: By induction on the recursion depth.

Base Case: Recursion depth = 0:

a) is trivial, since $\Im = \Im_0$. (def. 4.3.6)

b) is trivial since there are no free variables in $\mathcal{F}$.

Induction Step: Let the recursion depth be greater than 0.

Let $\pi(\mathcal{F}, \mathbb{c}, \mathbb{p}, \mathfrak{S})$ be the actual call to π and let C be a basic context sort.

The induction hypothesis states: For every $\Im \in \mathfrak{S}$: a) $\Im_{PL}(\mathbb{c}_C)(\mathcal{C}_{0C}) = \mathcal{C}_C$ and

b) $\Im_{PL}(\mathbb{p}(x))(\mathcal{C}_{0C}) = \mathcal{P}(x)$

In order to show that the statement also holds for the next recursion step we must perform a case analysis according to the structure of $\mathcal{F}$ and analyze the corresponding translation rule.

The nontrivial cases for b) are the quantifications over context variables. The induction hypothesis is immediately applicable to the free context variables in $\mathcal{G}$. Therefore the remaining variable to be checked is the newly quantified variable. The proofs for the existentially quantified variables are the same as for the universally quantified variables.

The induction hypothesis is immediately applicable to the term and atomic cases.

Case $\mathcal{F} = \forall x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

E) $\pi(\forall x\ \mathcal{G}, c, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c, \mathbf{p}, \boldsymbol{\mathfrak{I}}')$ $\qquad \boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in D_{\mathcal{S}\mathcal{V}}\}$.

a) Since formulae are standardized apart, x does not occur in c and therefore the induction hypothesis can immediately be applied.

Case $\mathcal{F} = \forall x\ \mathcal{G}$ and x:'C→qC' is a context variable.

The translation rule is

F) $\pi(\forall x\ \mathcal{G}, c, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/c_C \circ x], \mathbf{p}[x/c_C], \boldsymbol{\mathfrak{I}}')$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'C→qC'}_{\mathcal{C}}\}$.

Let $\mathfrak{I}' = (\mathfrak{F}, \mathcal{V}', \mathcal{C}', \mathcal{P}') \in \boldsymbol{\mathfrak{I}}'$.

a) $\mathfrak{I}'_{PL}(c_C \circ x)\ \mathcal{C}_{0C}) = \mathfrak{I}'_{PL}(x)(\mathfrak{I}'_{PL}(c_C)(\mathcal{C}_{0C}))$
$\quad = \mathfrak{I}'_{PL}(x)(\mathfrak{I}_{PL}(c_C)(\mathcal{C}_{0C}))$ $\qquad$ $(x \notin c_C)$
$\quad = \mathfrak{I}'_{PL}(x)(\mathcal{C}_C))$ $\qquad$ (induction hypothesis a)
$\quad = \chi(\mathcal{C}_C)$ $\qquad$ $(\mathfrak{I}'_{PL}(x) = \chi)$
$\quad = \mathcal{C}'_C$.

b) $\mathfrak{I}'_{PL}(\mathbf{p}(x))(\mathcal{C}_{0C}) = \mathfrak{I}'_{PL}(c_C)(\mathcal{C}_{0C})$
$\quad = \mathcal{C}_C$ $\qquad$ (induction hypothesis a)
$\quad = \mathcal{P}(x)$

Case $\mathcal{F} = \forall x\text{-}y\ \mathcal{G}$ and x:'C→qC' is a context variable.

The translation rule is

G) $\pi(\forall x\text{-}y\ \mathcal{G}, c, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/\mathbf{p}(y) \circ x], \mathbf{p}[x/\mathbf{p}(y)], \boldsymbol{\mathfrak{I}}')$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'C→qC'}_{\mathcal{C}}\}$

Let $\mathfrak{I}' = (\mathfrak{F}, \mathcal{V}', \mathcal{C}', \mathcal{P}') \in \boldsymbol{\mathfrak{I}}'$.

a) $\mathfrak{I}'_{PL}(\mathbf{p}(y) \circ x)(\mathcal{C}_0) = \mathfrak{I}'_{PL}(x)(\mathfrak{I}'_{PL}(\mathbf{p}(y))(\mathcal{C}_{0C}))$
$\quad = \mathfrak{I}'_{PL}(x)(\mathfrak{I}_{PL}(\mathbf{p}(y))(\mathcal{C}_{0C}))$ $\qquad$ $(x \notin \mathbf{p}(y))$
$\quad = \mathfrak{I}'_{PL}(x)(\mathcal{P}(y))$ $\qquad$ (induction hypothesis b)
$\quad = \chi(\mathcal{P}(y))$ $\qquad$ $(\mathfrak{I}'_{PL}(x) = \chi)$
$\quad = \mathcal{C}'_C$.

b) $\mathfrak{I}'_{PL}(\mathbf{p}(x))(\mathcal{C}_{0C}) = \mathfrak{I}'_{PL}(\mathbf{p}(y))(\mathcal{C}_{0C})$
$\quad = \mathfrak{I}_{PL}(\mathbf{p}(y))(\mathcal{C}_{0C})$ $\qquad$ $(x \notin \mathbf{p}(y))$
$\quad = \mathcal{P}(y)$ $\qquad$ (induction hypothesis b)
$\quad = \mathcal{P}(x)$

Case $\mathcal{F} = \forall{\downarrow}(x,z=t)\ \mathcal{G}$ and x:'D,C→qC' is a context variable.

The translation rule is

H) $\pi(\forall{\downarrow}(x,z=t)\ \mathcal{G}, c, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/c_C \circ {\downarrow}(x, \pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}}))], \mathbf{p}[x/c_C], \boldsymbol{\mathfrak{I}}')[z \leftarrow \pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}})]$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'D,C→qC'}_{\mathcal{C}}\}$

Let $\mathfrak{I}' = (\mathfrak{F}, \mathcal{V}', \mathcal{C}', \mathcal{P}') \in \boldsymbol{\mathfrak{I}}'$.

a) $\mathfrak{I}'_{PL}(c_C \circ {\downarrow}(x, \pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}})))(\mathcal{C}_{0C}) = \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}})))(\mathfrak{I}'_{PL}(c_C)(\mathcal{C}_{0C}))$
$\quad = \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}})))(\mathfrak{I}_{PL}(c_C)(\mathcal{C}_{0C}))$ $\qquad$ $(x \notin c_C)$
$\quad = \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}})))\ (\mathcal{C}_C)$ $\qquad$ (induction hypothesis a)
$\quad = \chi(\mathfrak{I}'_{PL}(\pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}})))\ (\mathcal{C}_C)$ $\qquad$ $(\mathfrak{I}'_{PL}(x) = \chi)$
$\quad = \chi(\mathfrak{I}(t))\ (\mathcal{C}_C)$ $\qquad$ (lemma 4.3.8 and ind. hyp.)
$\quad = \chi(\mathfrak{I}(t), \mathcal{C}_C)$
$\quad = \mathcal{C}'_C$.

b) For the $\mathfrak{I}' \in \boldsymbol{\mathfrak{I}}'$ the proof is the same as in the previous case.
For the $\pi(t, c, \mathbf{p}, \boldsymbol{\mathfrak{I}})$-calls the induction hypothesis is immediately applicable.

Case $\mathcal{F}=\forall{\downarrow}(x,z=t)\text{-}y\ \mathcal{G}$ and x:'D,C→^{q}C' is a context variable.

The translation rule is

I) $\pi(\forall{\downarrow}(x,z=t)\text{-}y\ \mathcal{G}, \mathbb{c}, \mathbb{p}, \mathbb{I}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{p}(y)\circ{\downarrow}(x, \pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I}))], \mathbb{p}[x/\mathbb{p}(y)], \mathbb{I}')[z\leftarrow\pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I})]$

$\mathbb{I}' := \{\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \mid \mathfrak{I} \in \mathbb{I}, \chi\in \text{'D,C}\to{}^q\text{C'}_{\mathcal{C}}\}$.

Let $\mathfrak{I}' = (\mathcal{F}, \mathcal{V}, \mathcal{C}, \mathcal{P}) \in \mathbb{I}'$.

a) $\mathfrak{I}'_{PL}(\mathbb{p}(y)\circ{\downarrow}(x, \pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I})))(\mathcal{C}_{0C}) = \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I})))(\mathfrak{I}'_{PL}(\mathbb{p}(y))(\mathcal{C}_{0C}))$

$= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I})))(\mathfrak{I}_{PL}(\mathbb{p}(y))(\mathcal{C}_{0C}))$ ($x \notin \mathbb{p}(y)$)

$= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I})))\ (\mathcal{P}(y))$ (induction hypothesis b)

$= \chi(\mathfrak{I}'_{PL}(\pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I})))\ (\mathcal{P}(y))$ ($\mathfrak{I}'_{PL}(x) = \chi$)

$= \chi(\mathfrak{I}(t))\ (\mathcal{P}(y))$ (lemma 4.3.8 and ind. hyp.)

$= \chi(\mathfrak{I}(t), \mathcal{P}(y))$

$= \mathcal{C}'_{C}$.

b) $\mathfrak{I}'_{PL}(\mathbb{p}(x))(\mathcal{C}_{0C}) = \mathfrak{I}'_{PL}(\mathbb{p}(y))(\mathcal{C}_{0C})$

$= \mathfrak{I}_{PL}(\mathbb{p}(y))(\mathcal{C}_{0C})$ ($x \notin \mathbb{p}(y)$)

$= \mathcal{P}(y)$ (induction hypothesis b)

$= \mathcal{P}(x)$.

Case $\mathcal{F}= \exists x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

J) $\pi(\exists x\ \mathcal{G}, \mathbb{c}, \mathbb{p}, \mathbb{I}) := \pi(\mathcal{G}, \mathbb{c}, \mathbb{p}, \mathbb{I}')[x\leftarrow g(s_1,\ldots,s_k, y_1,\ldots,y_m)]$

$\mathbb{I}' := \{\mathfrak{I}'=\mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \mathbb{I}, \chi\in D_{\mathcal{SV}}, \mathfrak{I}' \models_C \mathcal{G}\}$.

Since x does not occur in c, the induction hypothesis can immediately be applied.

Case $\mathcal{F}= \exists x\ \mathcal{G}$ and x:'C→^{q}C' is a context variable.

The translation rule is

K) $\pi(\exists x\ \mathcal{G}, \mathbb{c}, \mathbb{p}, \mathbb{I}) := \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{c}_C\circ x], \mathbb{p}[x/\mathbb{c}_C], \mathbb{I}')[x\leftarrow h(s_1,\ldots,s_{j-1},s_{j+1},\ldots s_k, y_1,\ldots,y_m)]$

$\mathbb{I}' := \{\mathfrak{I}'=\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \mathbb{I}, \chi\in \text{'C}\to{}^q\text{C'}_{\mathcal{C}}, \mathfrak{I}' \models_C \mathcal{G}\}$

Let $\mathfrak{I}' = (\mathcal{F}, \mathcal{V}, \mathcal{C}, \mathcal{P}) \in \mathbb{I}'$.

a) $\mathfrak{I}'_{PL}(\mathbb{c}_C\circ x)(\mathcal{C}_{0C}) = \mathfrak{I}'_{PL}(x)(\mathfrak{I}'_{PL}(\mathbb{c}_C)(\mathcal{C}_{0C}))$

$= \mathfrak{I}'_{PL}(x)(\mathfrak{I}_{PL}(\mathbb{c}_C)(\mathcal{C}_{0C}))$ ($x \notin \mathbb{c}_C$)

$= \mathfrak{I}'_{PL}(x)(\mathcal{C}_C)$ (induction hypothesis a)

$= \chi(\mathcal{C}_C)$ ($\mathfrak{I}'_{PL}(x) = \chi$)

$= \mathcal{C}'_C$.

Case $\mathcal{F}= \exists x\text{-}y\ \mathcal{G}$ and x:'C→^{q}C' is a context variable.

The translation rule is

L) $\pi(\exists x\text{-}y\ \mathcal{G}, \mathbb{c}, \mathbb{p}, \mathbb{I}):=\pi(\mathcal{G}, \mathbb{c}[C/\mathbb{p}(y)\circ x], \mathbb{p}[x/\mathbb{p}(y)], \mathbb{I}')[x\leftarrow h(s_1,\ldots,s_{j-1},s_{j+1},\ldots s_k,y_1,\ldots,y_m)]$

$\mathbb{I}' := \{\mathfrak{I}'=\mathfrak{I}[x/\chi]_{\mathcal{V}}[C/\chi(\mathcal{P}(y))]_{\mathcal{C}}[x/\mathcal{P}(y)]_{\mathcal{P}} \mid \mathfrak{I} \in \mathbb{I}, \chi\in \text{'C}\to{}^q\text{C'}_{\mathcal{C}}, \mathfrak{I}' \models_C \mathcal{G}\}$

Let $\mathfrak{I}' = (\mathcal{F}, \mathcal{V}, \mathcal{C}, \mathcal{P}) \in \mathbb{I}'$.

a) $\mathfrak{I}'_{PL}(\mathbb{p}(y)\circ x)(\mathcal{C}_{0C}) = \mathfrak{I}'_{PL}(x)(\mathfrak{I}'_{PL}(\mathbb{p}(y))(\mathcal{C}_{0C}))$

$= \mathfrak{I}'_{PL}(x)(\mathfrak{I}_{PL}(\mathbb{p}(y))(\mathcal{C}_{0C}))$ ($x \notin \mathbb{p}(y)$)

$= \mathfrak{I}'_{PL}(x)(\mathcal{P}(y))$ (induction hypothesis b)

$= \chi(\mathcal{P}(y))$ ($\mathfrak{I}'_{PL}(x) = \chi$)

$= \mathcal{C}'_C$.

Case $\mathcal{F}= \exists{\downarrow}(x,z=t)\ \mathcal{G}$ and x:'D,C→^{q}C' is a context variable.

The translation rule is

M) $\pi(\exists{\downarrow}(x,z=t)\ \mathcal{G}, \mathbb{c}, \mathbb{p}, \mathbb{I})$

$:= \pi(\mathcal{G}, \mathbb{c}[C/\mathbb{c}_C\circ{\downarrow}(x, \pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I}))], \mathbb{p}[x/\mathbb{c}_C], \mathbb{I}')\ [x\leftarrow k(s_1,\ldots,s_{j-1},s_{j+1},\ldots s_k,y_1,\ldots,y_m), z\leftarrow\pi(t, \mathbb{c}, \mathbb{p}, \mathbb{I})]$

$\mathbb{I}' := \{\mathfrak{I}'= \mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \mathbb{I}, \chi\in \text{'D,C}\to{}^q\text{C'}_{\mathcal{C}}, \mathfrak{I}' \models_C \mathcal{G}\}$

Let $\mathfrak{I}' = (\mathfrak{F}, \mathcal{V}', \mathcal{C}', \mathcal{P}') \in \mathbb{I}'$.

$$\begin{aligned}
\text{a)}\ \mathfrak{I}'_{PL}(\mathfrak{c}_C \circ {\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))(\mathcal{C}_{0C}) &= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))(\mathfrak{I}'_{PL}(\mathfrak{c}_C)(\mathcal{C}_{0C})) \\
&= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))(\mathfrak{I}_{PL}(\mathfrak{c}_C)(\mathcal{C}_{0C})) && (x \notin \mathfrak{c}_C) \\
&= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))\ (\mathcal{C}_C) && (\text{induction hypothesis a)}) \\
&= \chi(\mathfrak{I}'_{PL}(\pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))\ (\mathcal{C}_C) && (\mathfrak{I}'_{PL}(x) = \chi) \\
&= \chi(\mathfrak{I}(t))\ (\mathcal{C}_C) && (\text{lemma 4.3.8 and ind. hyp.}) \\
&= \chi(\mathfrak{I}(t), \mathcal{C}_C) \\
&= \mathcal{C}'_C.
\end{aligned}$$

Case $\mathcal{F} = \exists{\downarrow}(x,z{=}t)\text{-}y\ \mathcal{G}$ and x:'D,C→^qC' is a context variable.

The translation rule is

N) $\pi(\exists{\downarrow}(x,z{=}t)\text{-}y\ \mathcal{G}, \mathfrak{c}, \mathfrak{p}, \mathbb{I})$

$:= \pi(\mathcal{G}, \mathfrak{c}[C/\mathfrak{p}(x)\circ{\downarrow}(x, \pi(t,\mathfrak{c},\mathfrak{p},\mathbb{I}))], \mathfrak{p}[x/\mathfrak{p}(x)], \mathbb{I}')[x \leftarrow k(s_1,\ldots,s_{j-1},s_{j+1},\ldots s_k,y_1,\ldots,y_m), z \leftarrow \pi(t,\mathfrak{c},\mathfrak{p},\mathbb{I})]$

$\mathbb{I}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{P}(x))]_{\mathcal{C}}[x/\mathcal{P}(x)]_{\mathcal{P}} \mid \mathfrak{I} \in \mathbb{I}, \chi \in \text{'D,C}\rightarrow^q\text{C'}_{\mathcal{C}}\ \mathfrak{I}' \models_C \mathcal{G}\}$

Let $\mathfrak{I}' = (\mathfrak{F}, \mathcal{V}', \mathcal{C}', \mathcal{P}') \in \mathbb{I}'$.

$$\begin{aligned}
\text{a)}\ \mathfrak{I}'_{PL}(\mathfrak{c}_C \circ {\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))(\mathcal{C}_{0C}) &= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))(\mathfrak{I}'_{PL}(\mathfrak{p}(x))(\mathcal{C}_{0C})) \\
&= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))(\mathfrak{I}_{PL}(\mathfrak{p}(x))(\mathcal{C}_{0C})) && (x \notin \mathfrak{c}_C) \\
&= \mathfrak{I}'_{PL}({\downarrow}(x, \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))\ (\mathcal{P}(x)) && (\text{induction hypothesis b)}) \\
&= \chi(\mathfrak{I}'_{PL}(\pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})))\ (\mathcal{P}(x)) && (\mathfrak{I}'_{PL}(x) = \chi) \\
&= \chi(\mathfrak{I}(t))\ (\mathcal{P}(x)) && (\text{lemma 4.3.8 and ind. hyp.}) \\
&= \chi(\mathfrak{I}(t), \mathcal{P}(x)) \\
&= \mathcal{C}'_C.
\end{aligned}$$

Case $\mathcal{F} = \wp\ t\ \mathcal{G}$ and t:'C→^qC' is a context access term.

The translation rule is

O) $\pi(\wp\ t\ \mathcal{G}, \mathfrak{c}, \mathfrak{p}, \mathbb{I}) := \pi(\mathcal{G}, \mathfrak{c}[C/\mathfrak{c}_C \circ \pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I})], \mathfrak{p}, \mathbb{I}')\quad \mathbb{I}' := \{\mathfrak{I}[C/\mathfrak{I}(t)(\mathcal{C}_C)]_{\mathcal{C}} \mid \mathfrak{I} \in \mathbb{I}\}$

Let $\mathfrak{I}' \in \mathbb{I}'$.

$$\begin{aligned}
\text{a)}\ \mathfrak{I}'_{PL}(\mathfrak{c}_C \circ t)(\mathcal{C}_{0C}) &= \mathfrak{I}'_{PL}(\pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I}))(\mathfrak{I}_{PL}(\mathfrak{c}_C)(\mathcal{C}_{0C})) \\
&= \mathfrak{I}'_{PL}(\pi(t, \mathfrak{c}, \mathfrak{p}, \mathbb{I}))\ (\mathcal{C}_C) && (\text{induction hypothesis a)}) \\
&= \mathfrak{I}(t)\ (\mathcal{C}_C) && (\text{lemma 4.3.8 and ind. hyp.}) \\
&= \mathcal{C}'_C.
\end{aligned}$$

Cases $\mathcal{F} = \mathcal{G}_1 \wedge \mathcal{G}_2$ and $\mathcal{F} = \mathcal{G}_1 \vee \mathcal{G}_2$. These cases are straightforward. ■

Lemma 4.3.10 The Soundness Lemma

If $\mathfrak{I}_0 = (\mathfrak{F}, \mathcal{V}_0, \mathcal{C}_0, \mathcal{P}_0)$ is a CL-interpretation satisfying a CL-specification $\mathcal{S}$ then for each recursive call $\pi(\mathcal{F}, \mathfrak{c}, \mathfrak{p}, \mathbb{I})$ during the translation of the formulae in $\mathcal{S}$, the following invariant holds as long as $\mathcal{F}$ is a formula:

For every $\mathfrak{I} = (\mathfrak{F}, \mathcal{V}, \mathcal{C}, \mathcal{P}) \in \mathbb{I}$: $\mathfrak{I}_{PL} \models_P \pi(\mathcal{F}, \mathfrak{c}, \mathfrak{p}, \mathbb{I})$ where $\mathfrak{I}_{PL} := \Pi_3(\mathfrak{I})$

Proof: By induction on the structure of $\mathcal{G}$.

First Base Case:

$\mathcal{F} = \pm P(t_1,\ldots,t_n)$ where P is a domain predicate.

The translation rule is

C) $\pi(\pm P(t_1,\ldots,t_n), \mathfrak{c}, \mathfrak{p}, \mathbb{I}) := \pm\Pi_\Sigma(P)(s_1,\ldots,s_k, \pi(t_1, \mathfrak{c}, \mathfrak{p}, \mathbb{I}),\ldots, \pi(t_n, \mathfrak{c}, \mathfrak{p}, \mathbb{I}))$

$s_i = p_i({\downarrow}(\mathfrak{c}_{Ci}, 0_{Ci}))$ where $p_i{:}C_i \rightarrow I_i$ is the i-th projector function in $\mathcal{SH}(f)$.

We consider the positive case, i.e. $\mathcal{F} = P(t_1,\ldots,t_n)$ first.

Let $\mathfrak{I} \in \mathbb{I}$ and $\mathfrak{I}_{PL} =: (\mathcal{M}, \mathcal{V})$

$$\begin{aligned}
&\mathfrak{I}_{PL} \models_P \pi(P(t_1,\ldots,t_n), \mathfrak{c}, \mathfrak{p}, \mathbb{I}) \\
&\quad\text{iff } \mathfrak{I}_{PL}(s_1,\ldots,s_k, \pi(t_1, \mathfrak{c}, \mathfrak{p}, \mathbb{I}),\ldots, \pi(t_n, \mathfrak{c}, \mathfrak{p}, \mathbb{I})) \in P_{\mathcal{M}} && (\text{def. 4.2.5}) \\
&\quad\text{iff } (\mathfrak{I}_{PL}(s_1),\ldots,\mathfrak{I}_{PL}(s_k), \mathfrak{I}_{PL}(\pi(t_1, \mathfrak{c}, \mathfrak{p}, \mathbb{I})),\ldots, \mathfrak{I}_{PL}(\pi(t_n, \mathfrak{c}, \mathfrak{p}, \mathbb{I}))) \in P_{\mathcal{M}} \\
&\quad\text{iff } (\mathfrak{I}_{PL}(s_1),\ldots,\mathfrak{I}_{PL}(s_k), \mathfrak{I}(t_1),\ldots, \mathfrak{I}(t_n)) \in P_{\mathcal{M}} && (\text{lemma 4.3.8, 4.3.9}) \\
&\quad\text{iff } (\mathfrak{I}_{PL}(p_1)(\mathcal{C}_{C1}),\ldots,\mathfrak{I}_{PL}(p_k)(\mathcal{C}_{Ck}), \mathfrak{I}(t_1),\ldots, \mathfrak{I}(t_n)) \in P_{\mathcal{M}} && (\text{lemma 4.3.9})
\end{aligned}$$

iff $(\mathfrak{I}(t_1),\ldots, \mathfrak{I}(t_n))) \in P_{\mathcal{A}}$ $\mathcal{A} := \mathcal{SV}(\mathfrak{I}_{PL}(p_1)(\mathcal{C}_{C1}),\ldots,\mathfrak{I}_{PL}(p_k)(\mathcal{C}_{Ck}))$ (def. 4.3.4,3)
$\mathcal{A}$ is the actual Σ-structure for P (def. 4.2.3)
iff $\mathfrak{I} \models_C \mathcal{F}$ (def. 4.2.5)
iff true (lemma 4.3.7)

The negative case is analogous.

Second Base Case:

$\mathcal{F} = \pm P(t_1,\ldots,t_n)$ where P is a context predicate.

The translation rule is

D) $\pi(\pm P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}, \mathfrak{S}) := \pm P(s_1,\ldots,s_n)$

The s_i are determined as follows:

If t_i:'C→^{q}C' is a context variable x or a term $\downarrow(x,z)$ then $s_i := \downarrow((\mathbb{p}(x)\circ t_i)), 0_C)$.

If $t_i = s \circ t$ and s is a variable x:'C→^{q}C' or a term $\downarrow(x,z)$ then $s_i := \downarrow((\mathbb{p}(x)\circ s\circ\pi(t, \mathbb{c}, \mathbb{p})), 0_C)$,

otherwise $s_i := \pi(t_i, \mathbb{c}, \mathbb{p})$.

Again we consider the positive case first.

Let $\mathfrak{I} \in \mathfrak{S}$ and $\mathfrak{I}_{PL} =: (\mathcal{M}, \mathcal{V})$

$\mathfrak{I}_{PL} \models_P \pi(P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}, \mathfrak{S})$

iff $\mathfrak{I}_{PL}(s_1,\ldots,s_n) \in P_{\mathcal{M}}$ (def. 4.2.5)

iff $(\mathfrak{I}_{PL}(s_1),\ldots,\mathfrak{I}_{PL}(s_n)) \in P_{\mathcal{M}}$

We consider the above cases for t_i separately.

Case 1: t_i:'C→^{q}C' is either a context variable x or a term $\downarrow(x,z)$ and $s_i = \downarrow((\mathbb{p}(x)\circ t_i)), 0_C)$

$\mathfrak{I}_{PL}(s_i) = \mathfrak{I}_{PL}(t_i)(\mathfrak{I}_{PL}(\mathbb{p}(x)))\ (\mathfrak{I}_{PL}(0_C))$
$= \mathfrak{I}_{PL}(t_i)(\mathfrak{I}_{PL}(\mathbb{p}(x)))\ (\mathcal{C}_{0C})$ (def. 4.3.4,2)
$= \mathfrak{I}_{PL}(t_i)(\mathcal{P}(x))$ (lemma 4.3.9,b)
$= \mathfrak{I}(t_i)\ (\mathcal{P}(x))$ (lemma 4.3.8, 4.3.9)

Case 2: $t_i = s \circ t$ and s is a variable x:'C→^{q}C' or a term $\downarrow(x,z)$ and $s_i := \downarrow((\mathbb{p}(x)\circ s\circ\pi(t, \mathbb{c}, \mathbb{p}, \mathfrak{S})), 0_C)$

$\mathfrak{I}_{PL}(s_i) = \mathfrak{I}_{PL}(\pi(t, \mathbb{c}, \mathbb{p}, \mathfrak{S}))(\mathfrak{I}_{PL}(s)\ (\mathfrak{I}_{PL}(\mathbb{p}(x)))\ (\mathfrak{I}_{PL}(0_C)))$
$= \mathfrak{I}_{PL}(\pi(t, \mathbb{c}, \mathbb{p}, \mathfrak{S}))(\mathfrak{I}_{PL}(s)\ (\mathfrak{I}_{PL}(\mathbb{p}(x)))\ (\mathcal{C}_{0C})$ (def. 4.3.4,2)
$= \mathfrak{I}_{PL}(\pi(t, \mathbb{c}, \mathbb{p}, \mathfrak{S}))(\mathfrak{I}_{PL}(s)\ (\mathcal{P}(x)))$ (lemma 4.3.9,b)
$= \mathfrak{I}_{PL}(\pi(t, \mathbb{c}, \mathbb{p}, \mathfrak{S}))(\mathfrak{I}(s)\ (\mathcal{P}(x)))$ (lemma 4.3.8, 4.3.9)
$= \mathfrak{I}(t)\ (\mathfrak{I}(s)\ (\mathcal{P}(x))$ (lemma 4.3.8, 4.3.9)

Case 3: $s_i := \pi(t_i, \mathbb{c}, \mathbb{p}, \mathfrak{S})$.

$\mathfrak{I}_{PL}(s_i) = \mathfrak{I}(t_i)$ (lemma 4.3.8, 4.3.9)

Since $\mathfrak{I} \models_C \mathcal{F}$ (lemma 4.3.7)

and since the above three cases confirmed the three conditions in def. 4.2.5, we conclude

$(\mathfrak{I}_{PL}(s_1),\ldots,\mathfrak{I}_{PL}(s_n)) \in P_C$

and since according to def.4.1.1,2ii, the $\mathfrak{I}_{PL}(s_i)$ are not functional, we can apply def. 4.3.4,1and get the desired relation $(\mathfrak{I}_{PL}(s_1),\ldots,\mathfrak{I}_{PL}(s_n)) \in P_{\mathcal{M}}$, thus, $\mathfrak{I}_{PL} \models_P \pi(P(t_1,\ldots,t_n), \mathbb{c}, \mathbb{p}, \mathfrak{S})$

Induction Step: Let $\mathcal{F}$ be a formula, but no literal and let $\mathcal{F}_P := \pi(\mathcal{F}, \mathbb{c}, \mathbb{p}, \mathfrak{S})$ be the translated formula.

With lemma 4.3.7 we know $\forall\, \mathfrak{I} \in \mathfrak{S}: \mathfrak{I} \models_C \mathcal{F}$.

The induction hypothesis states for all recursive calls $\pi(\mathcal{F}', \mathbb{c}', \mathbb{p}', \mathfrak{S}')$ inside π:

$\forall\, \mathfrak{I}' \in \mathfrak{S}': \mathfrak{I}'_{PL} \models_P \pi(\mathcal{F}', \mathbb{c}', \mathbb{p}', \mathfrak{S}')$

In order to show $\forall\, \mathfrak{I} \in \mathfrak{S}: \mathfrak{I}_{PL} \models_P \mathcal{F}_P$ we perform a case analysis according to the structure of $\mathcal{F}$.

Case $\mathcal{F} = \forall x\ \mathcal{G}$ where x:D is a domain variable.

The translation rule is

E) $\pi(\forall x\ \mathcal{G}, \mathbb{c}, \mathbb{p}, \mathfrak{S}) := \forall x\ \pi(\mathcal{G}, \mathbb{c}, \mathbb{p}, \mathfrak{S}')$ $\mathfrak{S}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \mathfrak{S}, \chi \in D_{\mathcal{SV}}\}$.

$\forall\, \mathfrak{I} \in \mathfrak{S}: \mathfrak{I}_{PL} \models_P \mathcal{F}_P$ follows immediately from the induction hypothesis and def. 3.3.3.

Cases $\mathcal{F} = \forall x\ \mathcal{G}$ and $\forall x\text{-}y\ \mathcal{G}$ where x:'C→^{q}C' is a context variable.

The argument is the same as in the previous case (The $\mathcal{C}$ and $\mathcal{P}$-components play no role).

Case $\mathcal{F} = \forall\downarrow(x,z=t)\ \mathcal{G}$ where x:'D,C→ᑫC' is a context variable.

The translation rule is

H) $\pi(\forall\downarrow(x,z=t)\ \mathcal{G}, c, p, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, c[C/c_C\circ\downarrow(x, \pi(t, c, p, \boldsymbol{\mathfrak{I}}))], p[x/c_C], \boldsymbol{\mathfrak{I}}')[z\leftarrow\pi(t, c, p, \boldsymbol{\mathfrak{I}})]$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'D,C}\rightarrow{}^qC\text{'}_{\mathcal{C}}\}$

Let $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$. From the induction hypothesis and def. 3.3.3 we conclude

$\mathfrak{I}_{PL} \models_P \forall x\ \pi(\mathcal{G}, c[C/c_C\circ\downarrow(x, \pi(t, c, p, \boldsymbol{\mathfrak{I}}))], p[x/c_C], \boldsymbol{\mathfrak{I}}')$.

From lemma 4.3.8 we know $\mathfrak{I}(t) = \mathfrak{I}_{PL}(\pi(t, c, p, \boldsymbol{\mathfrak{I}}))$. Since $\mathfrak{I}'(z) = \mathfrak{I}(t)$ for every $\mathfrak{I}' \in \boldsymbol{\mathfrak{I}}$ we can replace z by $\pi(t, c, p, \boldsymbol{\mathfrak{I}})$ without changing the interpretation of any term. Therefore we finally get $\mathfrak{I}_{PL} \models_P \mathcal{F}_P$.

Case $\mathcal{F} = \forall\downarrow(x,z=t)\text{-}y\ \mathcal{G}$ where x:'D,C→ᑫC' is a context variable.

The argument is the same as in the previous case.

Case $\mathcal{F} = \exists x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

J) $\pi(\exists x\ \mathcal{G}, c, p, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c, p, \boldsymbol{\mathfrak{I}}')[x\leftarrow g(s_1,\ldots,s_k, y_1,\ldots,y_l)]$

Among the possible interpretations for g we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in D_{\mathcal{S}\mathcal{V}}$ with $\mathfrak{I}[x/\chi]_{\mathcal{V}} \models_C \mathcal{G}$

there is an χ' with $\mathfrak{I}_{PL}(g(s_1,\ldots,s_k, y_1,\ldots,y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in D_{\mathcal{S}\mathcal{V}}, \mathfrak{I}' \models_C \mathcal{G}\}$.

Let $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$.

From the induction hypothesis we know $\mathfrak{I}_{PL}[x/\chi'] \models_P \pi(\mathcal{G}, c, p, \boldsymbol{\mathfrak{I}}')$ for the object χ' mentioned in the semantic definition for g. Therefore χ' and the interpretation of $g(s_1,\ldots,s_k, y_1,\ldots,y_m)$ are identical and we can replace x by $g(s_1,\ldots,s_k, y_1,\ldots,y_m)$ without changing the interpretation of any term. Thus, $\mathfrak{I}_{PL}[x/\chi'] \models_P \pi(\mathcal{G}, c, p, \boldsymbol{\mathfrak{I}}')[x\leftarrow g(c_1,\ldots,c_k, y_1,\ldots,y_m)]$ and since x no longer occurs in the translated formula, $\mathfrak{I}_{PL} \models_P \mathcal{F}_P$.

Cases $\mathcal{F} = \exists x\ \mathcal{G}$ and $\mathcal{F} = \exists x\text{-}y\ \mathcal{G}$ and x:'C→ᑫC' is a context variable.

These cases are simplified versions of the next one.

Case $\mathcal{F} = \exists\downarrow(x,z=t)\ \mathcal{G}$ and x:'D,C→ᑫC' is a context variable.

The translation rule is

M) $\pi(\exists\downarrow(x,z=t)\ \mathcal{G}, c, p, \boldsymbol{\mathfrak{I}})$

$:= \pi(\mathcal{G}, c[C/c_C\circ\downarrow(x, \pi(t, c, p, \boldsymbol{\mathfrak{I}}))], p[x/c_C], \boldsymbol{\mathfrak{I}}')\ [x\leftarrow k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k, y_1,\ldots,y_m), z\leftarrow\pi(t,c,p,\boldsymbol{\mathfrak{I}})]$

Among the possible interpretations for k we select one satisfying the following condition:

For every $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$: Among the $\chi \in$ 'C→ᑫC'$_{\mathcal{C}}$ with $\mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[x/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x/\mathcal{C}_C]_{\mathcal{P}} \models_C \mathcal{G}$

there is a χ' with $\mathfrak{I}_{PL}(k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)) = \chi'$

$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}' = \mathfrak{I}[x/\chi, z/\mathfrak{I}(t)]_{\mathcal{V}}[C/\chi(\mathfrak{I}(t), \mathcal{C}_C)]_{\mathcal{C}}[x\ \mathcal{C}_C]_{\mathcal{P}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'D,C}\rightarrow{}^qC\text{'}_{\mathcal{C}}, \mathfrak{I}' \models_C \mathcal{G}\}$.

Let $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$. From the induction hypothesis we know $\mathfrak{I}_{PL}[x/\chi'] \models_P \pi(\mathcal{G}, c', \xi', \boldsymbol{\mathfrak{I}}')$ for the element χ' mentioned in the semantic definition for k. Therefore χ' and the interpretation of $k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k, y_1,\ldots,y_m)$ are identical. From lemma 4.3.8 we know that $\mathfrak{I}(t) = \mathfrak{I}_{PL}(\pi(t, c, p, \boldsymbol{\mathfrak{I}}))$. Since, according to the definition of domain formula, def. 4.1.3,iv x occurs only in the term $\downarrow(x,z)$ we can simultaneously replace x by $k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)$ and z by $\pi(t, c, p, \boldsymbol{\mathfrak{I}})$ without changing the interpretation of any term. Thus, $\mathfrak{I}_{PL}[x/\chi'] \models_P \mathcal{F}_P$ and since x no longer occurs in the translated formula, $\mathfrak{I}_{PL} \models_P \mathcal{F}_P$.

Case $\mathcal{F} = \exists\downarrow(x,z=t)\ \mathcal{G}$ and x:'D,C→ᑫC' is a context variable. The argument is the same as in the previous case.

Case $\mathcal{F} = \wp\ t\ \mathcal{G}$ and t:'C→ᑫC' is a context access term.

The translation rule is

O) $\pi(\wp\ t\ \mathcal{G}, c, p, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, c[C/c_C\circ\ \pi(t, c, p, \boldsymbol{\mathfrak{I}})], p, \boldsymbol{\mathfrak{I}}')\qquad \boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[C/\mathfrak{I}(t)(\mathcal{C}_C)]_{\mathcal{C}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$

Let $\mathfrak{I} \in \boldsymbol{\mathfrak{I}}$. Since $\mathfrak{I} \models_C \wp\ t\ \mathcal{G}, \mathfrak{I}' := \mathfrak{I}[C/\mathfrak{I}(t)(\mathcal{C}_C)]_{\mathcal{C}} \models_C \mathcal{G}$ and $\mathfrak{I}' \in \boldsymbol{\mathfrak{I}}'$ (def. 4.2.5)

$\Rightarrow\ \mathfrak{I}'_{PL} \models_P \pi(\mathcal{G}, c[C/c_C\circ\ \pi(t, c, p, \boldsymbol{\mathfrak{I}})], p, \boldsymbol{\mathfrak{I}}')$ (induction hypothesis)

Since $\mathfrak{I}'_{PL} = \mathfrak{I}_{PL}$, we conclude $\mathfrak{I}_{PL} \models_P \mathcal{F}_P$.

The remaining cases with ∧ and ∨ are straightforward. ■

Applied to the toplevel call of π, this lemma confirms that satisfiable CL-formulae are translated into satisfiable OSPL-formulae.

Completeness of the Translation

In the last section of chapter 4 we show that OSPL-satisfiability of a translated specification implies CL-satisfiability of the original CL-specification. We use $\Pi_{\mathfrak{I}}^{-1}$ to generate from the OSPL-model the corresponding CL-model. The completeness proof uses the same technique as the soundness proof, but just in the opposite direction. That means this time we follow the "top down" recursion of π to decompose the OSPL-model $\mathfrak{I}$ into the interpretations for the translated subformulae $\mathcal{G}_P$ in $\mathcal{F}_P$. When we have reached the atomic level we can translate the OSPL-interpre- tations for the OSPL-atoms into CL-interpretations for the original atoms and from these build up the CL-model for the original formula. To this end we redefine the augmented formula morphism $\Pi_{\mathcal{F}}$(def. 4.3.6) such that the additional argument takes an OSPL-model for the translated formula.

Notice that in the rest of the chapter $\mathfrak{I}$ always denotes an OSPL-interpretation whereas $\mathfrak{I}_{CL}$ denotes a CL-interpretation.

Definition 4.3.11 (The Redefined Augmented Formula Morphism)
We define the augmented formula morphism in a similar way as in definition 4.3.6, but reinterpret the $\boldsymbol{\mathfrak{I}}$-argument in π. Each element in $\boldsymbol{\mathfrak{I}}$ is now an OSPL-interpretation.

The toplevel call for π is again:
$\Pi(\mathcal{F}, \mathfrak{I}) = \pi(\mathcal{F}, \mathbf{c}_0, \mathbf{p}_0, \{\mathfrak{I}\})$ where $\mathbf{c}_0$ and $\mathbf{p}_0$ are the default values of def. 4.3.2.

The translation rules A, B, C, D, E and P are the same as in def. 4.3.6. The modified translation rules are:

F) $\pi(\forall x\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{c}_C \circ x], \mathbf{p}[x/\mathbf{c}_C], \boldsymbol{\mathfrak{I}}')$ where x:'C→^{q}C' is a context variable.
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'}C \rightarrow {}^{q}C\text{'}_{\mathcal{M}}\}$.

G) $\pi(\forall x\text{-}y\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{p}(y) \circ x], \mathbf{p}[x/\mathbf{p}(y)], \boldsymbol{\mathfrak{I}}')$ where x:'C→^{q}C' is a context variable.
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'}C \rightarrow {}^{q}C\text{'}_{\mathcal{M}}\}$.

H) $\pi(\forall{\downarrow}(x,z=t)\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{c}_C \circ {\downarrow}(x, \pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}))], \mathbf{p}[x/\mathbf{c}_C], \boldsymbol{\mathfrak{I}}')\ [z \leftarrow \pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}})]$
where x:'D,C→^{q}C' is a context variable.
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi, z/\mathfrak{I}(\pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}))]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'}D,C \rightarrow {}^{q}C\text{'}_{\mathcal{M}}\}$

I) $\pi(\forall{\downarrow}(x,z=t)\text{-}y\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \forall x\ \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{p}(y) \circ {\downarrow}(x, \pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}))], \mathbf{p}[x/\mathbf{p}(y)], \boldsymbol{\mathfrak{I}}')\ [z \leftarrow \pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}})]$
where x:'D,C→^{q}C' is a context variable.
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\chi, z/\mathfrak{I}(\pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}))]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}, \chi \in \text{'}D,C \rightarrow {}^{q}C\text{'}_{\mathcal{M}}\}$

J) $\pi(\exists x\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}')[x \leftarrow g(s_1,\ldots,s_k, y_1,\ldots,y_m)]$ where x:D is a domain variable.
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}[x/\mathfrak{I}(g(s_1,\ldots,s_k, y_1,\ldots,y_m))]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$

K) $\pi(\exists x\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{c}_C \circ x], \mathbf{p}[x/\mathbf{c}_C], \boldsymbol{\mathfrak{I}}')[x \leftarrow h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)]$
where x:'C→^{q}C' is a context variable.
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}'=\mathfrak{I}[x/\mathfrak{I}(h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m))]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$

L) $\pi(\exists x\text{-}y\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{p}(y) \circ x], \mathbf{p}[x/\mathbf{p}(y)], \boldsymbol{\mathfrak{I}}')[x \leftarrow h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)]$
where x:'C→^{q}C' is a context variable.
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}'=\mathfrak{I}[x/\mathfrak{I}(h(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k, y_1,\ldots,y_m))]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$

M) $\pi(\exists{\downarrow}(x,z=t)\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}})$ where x:'D,C→^{q}C' is a context variable.
$:= \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{c}_C \circ {\downarrow}(x, \pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}))], \mathbf{p}[x/\mathbf{c}_C], \boldsymbol{\mathfrak{I}}')[x \leftarrow k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_m,y_1,\ldots,y_l), z \leftarrow \pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}})]$
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}'=\mathfrak{I}[x/\mathfrak{I}(k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m)), z/\mathfrak{I}(\pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}))]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$

N) $\pi(\exists{\downarrow}(x,z=t)\text{-}y\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}})$ where x:'D,C→^{q}C' is a context variable.
$:= \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{p}(y) \circ {\downarrow}(x, \pi(t,\mathbf{c},\mathbf{p},\boldsymbol{\mathfrak{I}}))], \mathbf{p}[x/\mathbf{p}(y)], \boldsymbol{\mathfrak{I}}')[x \leftarrow k(s_1,\ldots,s_{j-1},s_{j+1},\ldots,s_k,y_1,\ldots,y_m), z \leftarrow \pi(t,\mathbf{c},\mathbf{p},\boldsymbol{\mathfrak{I}})]$
$\boldsymbol{\mathfrak{I}}' := \{\mathfrak{I}'=\mathfrak{I}[x/\mathfrak{I}(k(s_1,\ldots,s_{l-1},s_{l+1},\ldots,s_k,y_1,\ldots,y_l)), z/\mathfrak{I}(\pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}))]_{\mathcal{V}} \mid \mathfrak{I} \in \boldsymbol{\mathfrak{I}}\}$

O) $\pi(\wp\ t\ \mathcal{G}, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}, \mathbf{c}[C/\mathbf{c}_C \circ \pi(t, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}})], \mathbf{p}, \boldsymbol{\mathfrak{I}})$

Q) $\pi(\mathcal{G}_1 \vee \mathcal{G}_2, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}) := \pi(\mathcal{G}_1, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}_1) \vee \pi(\mathcal{G}_2, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}_2)$ $\boldsymbol{\mathfrak{I}}_i := \{\mathfrak{I} \in \boldsymbol{\mathfrak{I}} \mid \mathfrak{I} \models_P \pi(\mathcal{G}_i, \mathbf{c}, \mathbf{p}, \boldsymbol{\mathfrak{I}}_i)\}$ $i = 1,2$ ■

Again we have to verify that the decomposition of the OSPL-model is correct, i.e. that the generated interpretations in the $\mathfrak{S}$-argument of π really satisfy the corresponding translated subformula.

Lemma 4.3.12 The Redefined Augmented Formula Morphism is Well Defined.
If $\Im_0$ is a OSPL-interpretation satisfying a translated CL-specification $\mathcal{S}$ then for each recursive call $\pi(\mathcal{G}, c, \mathbb{p}, \mathfrak{S})$ during the translation of the formulae in $\mathcal{S}$, the following invariant holds as long as $\mathcal{G}$ is a formula:
For every $\Im \in \mathfrak{S}$: $\Im \models_P \mathcal{G}_P := \pi(\mathcal{G}, c, \mathbb{p}, \mathfrak{S})$.
Proof: By induction on the recursion depth.
Base Case: Recursion depth = 0: This is just the initial condition that $\Im_0$ satisfies the translated $\mathcal{S}$.
Induction Step: Let the recursion depth be greater than 0.
Let $\pi(\mathcal{F}, c, \mathbb{p}, \mathfrak{S})$ be the actual call to π and let $\mathcal{F}_P$ and $\mathcal{G}_P$ be the translated formulae.
The induction hypothesis states: For every $\Im \in \mathfrak{S}$: $\Im \models_P \mathcal{F}_P$.
In order to show that the statement also holds for the next recursion step we must perform a case analysis according to the structure of $\mathcal{F}$ and analyze the corresponding translation rule.
Case $\mathcal{F} = \forall x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

E) $\pi(\forall x\ \mathcal{G}, c, \mathbb{p}, \mathfrak{S}) := \forall x\ \pi(\mathcal{G}, c, \mathbb{p}, \mathfrak{S}')$ $\qquad$ $\mathfrak{S}' := \{\Im[x/\chi]_{\mathcal{V}} \mid \Im \in \mathfrak{S}, \chi \in D_{\mathcal{S}\mathcal{V}}\}$

The induction hypothesis immediately implies $\Im \in \mathfrak{S}'$: $\Im \models_P \mathcal{G}_P$. (def. 3.3.3)

The same holds for the remaining cases with universal quantifiers.
Case $\mathcal{F} = \exists x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

J) $\pi(\exists x\ \mathcal{G}, c, \mathbb{p}, \mathfrak{S}) := \pi(\mathcal{G}, c, \mathbb{p}, \mathfrak{S}')[x \leftarrow g(s_1,\ldots,s_k, y_1,\ldots,y_m)]$

$\mathfrak{S}' := \{\Im' = \Im[x/\Im(g(s_1,\ldots,s_k, y_1,\ldots,y_m))]_{\mathcal{V}} \mid \Im \in \mathfrak{S}\}$

Let $\Im' = \Im[x/\Im(g(s_1,\ldots,s_k, y_1,\ldots,y_m))]_{\mathcal{V}} \in \mathfrak{S}'$.

Since $\Im \models_P \mathcal{F}_P$ (induction hypothesis) and x is not in $\mathcal{F}_P$, $\Im' \models_P \mathcal{F}_P$.

Furthermore since $\Im'(x) = \Im(g(s_1,\ldots,s_k, y_1,\ldots,y_m))$, the interpretation of the terms in $\mathcal{F}_P$ remains the same when we replace $g(s_1,\ldots,s_k, y_1,\ldots,y_m)$ by x. Hence, $\Im' \models_P \pi(\mathcal{G}, c, \mathbb{p}, \mathfrak{S}')$.

The same holds for the remaining cases with existential quantifiers.
The proofs for the $\wp$, $\wedge$ and $\vee$ cases are straightforward. ∎

The definition of the inverse signature morphism (def. 4.3.4) tells us how models for translated CL-formulae are translated into CL-interpretation for the original CL-formula. This definition must be refined in order to get the right actual context $\mathcal{C}$ and context assignments $\mathcal{P}$ for the CL-interpretations which are obtained from the decomposed OSPL-interpretations in the redefined augmented formula morphism. The information from which these additional components can be generated is contained in the c- and $\mathbb{p}$-arguments of π: Since the special constant symbol 0_C denotes the initial C-context, applying the interpretation of c_C - a context access function - to this initial C-context yields the component $\mathcal{C}_C$. Application of the interpretation of $\mathbb{p}(x)$ - again a context access function - to the initial C-context on the other hand yields $\mathcal{P}(x)$.

Definition 4.3.13 The Associated CL-Interpretations
Let $\pi(\mathcal{F}, c, \mathbb{p}, \mathfrak{S})$ be a call to the redefined augmented formula morphism and let $C_1,\ldots,C_n$ be the basic context sorts. To each $\Im = (\mathcal{M}, \mathcal{V}) \in \mathfrak{S}$ we associate a CL-interpretation $\Im_{CL} := (\Pi_\Im^{-1}(\mathcal{M}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ where
$\mathcal{C} = (\Im(c_{C1})(0_{C1\mathcal{M}}),\ldots,\Im(c_{C1})(0_{C1\mathcal{M}}))$ and for the free context variables x of sort 'C→ᑫC' or 'D,C→ᑫC' in $\mathcal{F}$:
$\mathcal{P}(x) := \Im(\mathbb{p}(x))\ (0_{C1\mathcal{M}})$. ∎

Lemma 4.3.14 The Interpretation of Terms is Invariant

Let t be a domain term over a signature Σ, let $t_P := \pi(t, c, \mathbb{p}, \mathfrak{S})$ and for every $\mathfrak{I} = (\mathcal{M}, \mathcal{V}) \in \mathfrak{S}$ let $\mathfrak{I}_{CL} := (\Pi_{\mathfrak{I}}^{-1}(\mathcal{M}), \mathcal{V}, \mathcal{C}, \mathcal{P})$ according to def. 4.3.13.

Then for every $\mathfrak{I} \in \mathfrak{S}$: $\mathfrak{I}_{CL}(t) = \mathfrak{I}(t_P)$.

Proof: By induction on the structure of t.

Base Case: t is a variable. The statement is true because the translation neither changes variables nor their interpretation.

Induction Step: $t = f(t_1,\ldots,t_n)$.

$\mathfrak{I}(\pi(t, c, \mathbb{p}, \mathfrak{S})) = \mathfrak{I}(\Pi_\Sigma(f)(s_1,\ldots,s_k, \pi(t_1, c, \mathbb{p}, \mathfrak{S}),\ldots, \pi(t_n, c, \mathbb{p}, \mathfrak{S})))$

$s_i := p_i(\downarrow(c_{Ci}, 0_{Ci}))$ where $p_i{:}C_i \to I_i$ is the i-th element in $\mathcal{SV}(f)$ (def. 4.3.2,B)

$= f_{\mathcal{M}}(\mathfrak{I}(s_1),\ldots,\mathfrak{I}(s_k), \mathfrak{I}(\pi(t_1, c, \mathbb{p}, \mathfrak{S})),\ldots, \mathfrak{I}(\pi(t_n, c, \mathbb{p}, \mathfrak{S}))))$

$= f_{\mathcal{M}}(\mathfrak{I}(s_1),\ldots,\mathfrak{I}(s_k), \mathfrak{I}_{CL}(t_1),\ldots, \mathfrak{I}_{CL}(t_n)))$ (induction hypothesis)

$= f_{\mathcal{M}}(\mathfrak{I}(p_1)(\mathcal{C}_{C1}),\ldots,\mathfrak{I}(p_k)(\mathcal{C}_{C1}), \mathfrak{I}_{CL}(t_1),\ldots, \mathfrak{I}_{CL}(t_n)))$ (def. 4.3.13)

$= f_{\mathcal{A}}(\mathfrak{I}_{CL}(t_1),\ldots, \mathfrak{I}_{CL}(t_n)))$ $\quad \mathcal{A} := \mathcal{SV}(\mathfrak{I}(p_1)(\mathcal{C}_{C1}),\ldots,\mathfrak{I}(p_k)(\mathcal{C}_{C1}))$ (def. 4.3.4,2)

$\mathcal{A}$ is the actual Σ-structure for f. (def. 4.2.3)

$= \mathfrak{I}_{CL}(t)$ (def. 4.2.4) ■

Lemma 4.3.15 The Completeness Lemma

If $\mathfrak{I}_0$ is an OSPL-interpretation satisfying a translated specification $\Pi(\mathcal{S})$ then for each recursive call $\pi(\mathcal{F}, c, \mathbb{p}, \mathfrak{S})$ during the translation of the formulae in $\mathcal{S}$, the following invariant holds as long as $\mathcal{F}$ is a formula:

For every $\mathfrak{I} \in \mathfrak{S}$: $\mathfrak{I}_{CL} \models_C \mathcal{F}$.

Proof: By induction on the structure of $\mathcal{F}$.

First Base Case:

$\mathcal{F} = \pm P(t_1,\ldots,t_n)$ where P is a domain predicate.

The translation rule is

C) $\pi(\pm P(t_1,\ldots,t_n), c, \mathbb{p}, \mathfrak{S}) := \pm\Pi_\Sigma(P)(s_1,\ldots,s_k, \pi(t_1, c, \mathbb{p}, \mathfrak{S}),\ldots, \pi(t_n, c, \mathbb{p}, \mathfrak{S}))$

$s_i = p_i(\downarrow(c_{Ci}, 0_{Ci}))$ where $p_i{:}C_i \to I_i$ is the i-th projector function in $\mathcal{SV}(f)$.

We consider the positive case, i.e. $\mathcal{F} = P(t_1,\ldots,t_n)$ first.

Let $\mathfrak{I} \in \mathfrak{S}$ and $\mathfrak{I}_{CL} =: ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$

true iff (lemma 4.3.12)

$\mathfrak{I} \models_P \pi(P(t_1,\ldots,t_n), c, \mathbb{p}, \mathfrak{S})$

iff $\mathfrak{I}(c_1,\ldots,c_k, \pi(t_1, c, \mathbb{p}, \mathfrak{S}),\ldots, \pi(t_n, c, \mathbb{p}, \mathfrak{S})) \in P_{\mathcal{M}}$ (def. 4.2.5)

iff $(\mathfrak{I}(s_1),\ldots,\mathfrak{I}(s_k), \mathfrak{I}(\pi(t_1, c, \mathbb{p}, \mathfrak{S})),\ldots, \mathfrak{I}(\pi(t_n, c, \mathbb{p}, \mathfrak{S}))) \in P_{\mathcal{M}}$

iff $(\mathfrak{I}(s_1),\ldots,\mathfrak{I}(s_k), \mathfrak{I}_{CL}(t_1),\ldots, \mathfrak{I}_{CL}(t_n)) \in P_{\mathcal{M}}$ (lemma 4.3.14)

iff $(\mathfrak{I}(p_1)(\mathcal{C}_{C1}),\ldots,\mathfrak{I}(p_k)(\mathcal{C}_{Ck}), \mathfrak{I}_{CL}(t_1),\ldots, \mathfrak{I}_{CL}(t_n)) \in P_{\mathcal{M}}$ (def. 4.3.13)

iff $(\mathfrak{I}_{CL}(t_1),\ldots, \mathfrak{I}_{CL}(t_n))) \in P_{\mathcal{A}}$ $\quad \mathcal{A} := \mathcal{SV}(\mathfrak{I}(p_1)(\mathcal{C}_{C1}),\ldots,\mathfrak{I}(p_k)(\mathcal{C}_{C1}))$ (def 4.3.4,3)

$\mathcal{A}$ is the actual Σ-structure for P. (def. 4.2.3)

iff $\mathfrak{I}_{CL} \models_C \mathcal{F}$ (def. 4.2.5)

The negative case is analogous.

Second Base Case:

$\mathcal{F} = \pm P(t_1,\ldots,t_n)$ where P is a context predicate.

The translation rule is

D) $\pi(\pm P(t_1,\ldots,t_n), c, \mathbb{p}, \mathfrak{S}) := \pm P(s_1,\ldots,s_n)$

The s_i are determined as follows:

If t_i:'$C \to^q C$' is a context variable x or a term $\downarrow(x, z)$ then $s_i := \downarrow((\mathbb{p}(x) \circ t_i)), 0_C)$

If $t_i = s \circ t$ and x is a variable $x{:}C \to^q C$ or a term $\downarrow(x, z)$ then $s_i := \downarrow((\mathbb{p}(x) \circ s \circ \pi(t, c, \mathbb{p}, \mathfrak{S})), 0_C)$

otherwise $s_i := \pi(t_i, c, \mathbb{p}, \mathfrak{S})$.

Again we consider the positive case first.

Let $\Im \in \mathfrak{I}$ and $\Im_{CL} =: ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, \mathcal{C}, \mathcal{P})$.

Since according to lemma 4.3.12 $\Im \models_P \pi(P(t_1,\ldots,t_n), c, \mathbb{p}, \mathfrak{I})$ we know $(\Im(s_1),\ldots,\Im(s_n)) \in P_{\mathcal{M}}$.

Since according to def.4.1.1,2ii the $\Im(s_i)$ are not functional, we can apply def. 4.3.4,4a and get the relation $(\Im(s_1),\ldots,\Im(s_n)) \in P_{\mathcal{C}}$

We consider the above cases for t_i separately.

Case 1: t_i:'$C \to^q C$' is a context variable x or a term $\downarrow(x, z)$ and $s_i := \downarrow((\mathbb{p}(x) \circ t_i)), 0_C)$

$\Im(s_i)$	$= \Im(t_i)(\Im(\mathbb{p}(x)))\ (\Im(0_C))$	
	$= \Im(t_i)(\Im(\mathbb{p}(x)))\ (\mathcal{C}_0)$	(def. 4.3.4,2)
	$= \Im(t_i)(\mathcal{P}(x))$	(def. 4.3.13)
	$= \Im_{CL}(t_i)\ (\mathcal{P}(x))$	(lemma 4.3.14)

Case 2: $t_i = s \circ t$ and x is a variable $x{:}C \to^q C$ or a term $\downarrow(x, z)$ and $s_i := \downarrow((\mathbb{p}(x) \circ s \circ \pi(t, c, \mathbb{p}, \mathfrak{I})), 0_C)$

$\Im(s_i)$	$= \Im(\pi(t, c, \mathbb{p}, \mathfrak{I}))(\Im(s)\ (\Im\ \mathbb{p}(x)))\ (\Im(0_C)))$	
	$= \Im(\pi(t, c, \mathbb{p}, \mathfrak{I}))(\Im(s)\ (\Im(\mathbb{p}(x)))\ (\mathcal{C}_0)$	(def. 4.3.4,2)
	$= \Im(\pi(t, c, \mathbb{p}, \mathfrak{I}))(\Im(s)\ (\mathcal{P}(x)))$	(def. 4.3.13)
	$= \Im(\pi(t, c, \mathbb{p}, \mathfrak{I}))(\Im_{CL}(s)\ (\mathcal{P}(x)))$	(lemma 4.3.14)
	$= \Im_{CL}(t)\ (\Im_{CL}(s)\ (\mathcal{P}(x)))$	(lemma 4.3.14)

Case 3: $s_i := \pi(t_i, c, \mathbb{p}, \mathfrak{I})$.

$\Im(s_i)$	$= \Im_{CL}(t_i)$	(lemma 4.3.14)

Since the above three cases confirmed the three conditions in def. 4.2.5, we conclude $(\Im_{CL}(s_1),\ldots,\Im_{CL}(s_n)) \in P_{\mathcal{C}}$ i.e. $\Im_{CL} \models_C \mathcal{F}$.

Induction Step:

Let $\mathcal{F}$ be a formula, but no literal and let $\mathcal{F}_P := \pi(\mathcal{F}, c, \mathbb{p}, \mathfrak{I})$ be the translated formula.

With lemma 4.3.12 we know $\forall \Im \in \mathfrak{I}: \Im \models_P \mathcal{F}_P$.

The induction hypothesis states for all recursive calls $\pi(\mathcal{F}', c', \mathbb{p}', \mathfrak{I}')$ inside π: $\forall \Im' \in \mathfrak{I}': \Im'_{CL} \models_C \mathcal{F}'$

In order to show $\forall \Im \in \mathfrak{I}: \Im_{CL} \models_C \mathcal{F}$ we perform a case analysis according to the structure of $\mathcal{F}$.

Case $\mathcal{F} = \forall x\ \mathcal{G}$ where x:D is a domain variable.

The translation rule is

E) $\pi(\forall x\ \mathcal{G}, c, \mathbb{p}, \mathfrak{I}) := \forall x\ \pi(\mathcal{G}, c, \mathbb{p}, \mathfrak{I}')$ $\qquad \mathfrak{I}' := \{\Im[x/\chi]_{\mathcal{V}} \mid \Im \in \mathfrak{I}, \chi \in D_{\mathcal{SV}}\}$

$\forall \Im \in \mathfrak{I}: \Im_{CL} \models_C \mathcal{F}$ follows immediately form the induction hypothesis and def. 4.2.5.

The remaining cases with universal quantifiers are proved in the same way as the previous one.

Case $\mathcal{F} = \exists x\ \mathcal{G}$ and x:D is a domain variable.

The translation rule is

J) $\pi(\exists x\ \mathcal{G}, c, \mathbb{p}, \mathfrak{I}) := \pi(\mathcal{G}, c, \mathbb{p}, \mathfrak{I}')[x \leftarrow g(s_1,\ldots,s_k, y_1,\ldots,y_m)]$

$\mathfrak{I}' := \{\Im[x/\Im(g(s_1,\ldots,s_k, y_1,\ldots,y_m))]_{\mathcal{V}} \mid \Im \in \mathfrak{I}\}$.

Choosing $a = \Im(g(s_1,\ldots,s_k, y_1,\ldots,y_m))$, the statement follows again immediately from the induction hypothesis and def. 4.2.5.

The remaining cases with existential quantifiers are proved in the same way as the previous one.

The remaining cases with $\wp$, $\wedge$ and $\vee$ are straightforward. ∎

From the soundness and completeness lemmas we obtain now the final result of this chapter:

Theorem 4.3.16 Soundness and Completeness of the Translation

A Cl-specification $\mathcal{S}$ is Cl-satisfiable if and only if the translated OSPL-specification $\Pi(\mathcal{S})$ is OSPL-satisfiable.

This is an immediate consequence of the soundness and completeness lemmas 4.3.10 and 4.3.15. ∎

Chapter Five

Multi Modal Logic

The Context Logic methodology is designed for handling complex logics. In order to demonstrate what all the details are good for and how the various mechanisms cooperate we therefore need a nontrivial example. Therefore I have chosen a kind of multi modal logic (MM-Logic), an extension of first-order modal logic as an example logic. This extension is quite expressive and can serve as temporal, process, action and epistemic logic in various applications. The basis consists of the classical modal logics D, T, D4 and S4 with possible worlds semantics and it includes Clarke and Emerson's CTL temporal logic [Clarke&Emerson 83] as a fragment. The accessibility relation has to be serial, i.e. from each world there must be an accessible world. This is one basic assumption of CL. The other assumption is that the domains are identical in each world (constant-domain interpretations). However, we allow modal operators corresponding to accessibility relations with different properties to occur simultaneously. In particular we have a basic discrete accessibility relation, its reflexive, transitive and reflexive-transitive closure. The transitive closure is not completely axiomatizable in first-order logic, but we approximate it as far as possible.

The accessibility relations themselves can be labeled with arbitrary domain elements and we provide indexed operators which can refer to these labels. Furthermore we include an 'eventually' operator ◗ with the meaning ◗$\mathcal{F}$ is true in a world $\mathcal{W}$ if on every path $\mathcal{P}$ through the possible worlds structure starting with $\mathcal{W}$ there exists a world $\mathcal{W}_1$ such that $\mathcal{F}$ is true in $\mathcal{W}_1$. Finally we include 'until' opera- tors for accessing limited areas in the possible worlds structure. Hence, MM-Logic has two kinds of multiplicities, several accessibility relations simultaneously, and indexed operators. Function and predicate symbols are flexible, i.e. their interpretation may change from world to world. Extending this logic to deal with flexible and rigid designators simultaneously is a trivial exercise.

In the sequel let $\mathcal{R} \in \{\emptyset, r, t, rt,\}$ where '$\emptyset$' refers to the basic accessibility relation, 'r' refers to its reflexive, 't' to its transitive and rt to its reflexive-transitive closure. We do not consider symmetric accessibility relations because the interaction of the symmetric $\square^s$-operator with the other modal operators is utterly complicated.

The full set of logical connectives, quantifiers and operators we are going to use is

$\wedge$	(and)	$\forall$	(for all)
$\vee$	(or)	$\exists$	(there exists)
$\neg$	(not)	$\square^{\mathcal{R}}$	(necessarily)
$\Rightarrow$	(implies)	$\lozenge^{\mathcal{R}}$	(possibly)
$\Leftrightarrow$	(is equivalent)	◗	(eventually)
$[\![\ldots]\!]^{\mathcal{R}}$	(indexed necessarily)	$\twoheadrightarrow$	(possibly henceforth)
$<\ldots>^{\mathcal{R}}$	(indexed possibly)	$\forall U$, $\forall U^r$	(always until)
$\vert\ldots)$	(indexed eventually)	$\exists U$, $\exists U^r$	(possibly until)

The pairs ($\square^{\mathcal{R}}$, $\lozenge^{\mathcal{R}}$), (◗, $\twoheadrightarrow$), ($[\![\ldots]\!]^{\mathcal{R}}$, $<\ldots>^{\mathcal{R}}$) of operators are dual to each other. Duality means that moving a negation sign over one operator in that pair switches it to the other operator. For example $\neg\square\mathcal{F} \Leftrightarrow \lozenge\neg\mathcal{F}$. This property can be used to create a negation normal form for formulae by moving all negation signs in front of the atoms. Since, however, dual operators for $\vert\ldots)$ and the 'until' operators are not included, we can generate a negation normal form only for the translated Context Logic formulae.

Some of the operators are definable from others:

◗$\mathcal{F}$ $\Leftrightarrow$ true $\forall U$ $\mathcal{F}$ $\Leftrightarrow$ true $\forall U^r$ $\mathcal{F}$ and

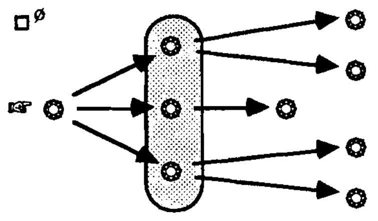
$\lozenge^{rt}\mathcal{F}$ $\Leftrightarrow$ true $\exists U$ $\mathcal{F}$ $\Leftrightarrow$ true $\exists U^r$ $\mathcal{F}$.

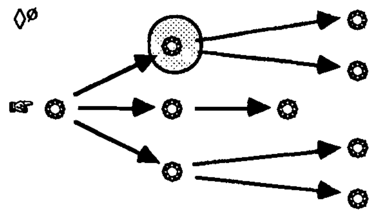
Nevertheless we treat them separately because the translation into OSPL can then be optimized.

Before defining syntax and semantics formally, let us first try to get some intuition about the meaning of the operators. As already said we assume a possible worlds structure with a basic accessibility relation together with some of their closures. The transitions from world to world may be labeled with a (possibly empty) set of domain elements. As a concrete interpretation of such a possible worlds structure, think of the worlds representing the current state of some interacting processes (software, hardware or whatsoever) and a transition indicating a single atomic action of a single process. The transition´s label is an identifier for the process that performed that action. We shall give other interpretations at the end of this chapter.

The figures below illustrate the effects of the operators. For an operator O and a formula $\mathcal{F}$ the marked worlds are those which have to verify $\mathcal{F}$ in order to verify $O\mathcal{F}$ in the actual world (which is labeled with ☞). I.e. the marked worlds are those which are in some sense accessed by the operator. The operator in the left figure is usually of universal force, whereas the operator in the right figure is the dual one, i.e. it is of existential force.

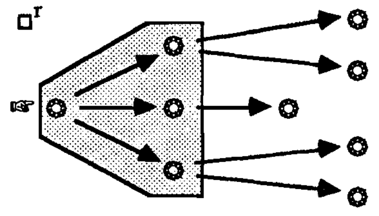
The operators $\Box^{\emptyset}$ (access to *all* directly accessible worlds) and
$\Diamond^{\emptyset}$ (access to *some* directly accessible worlds):

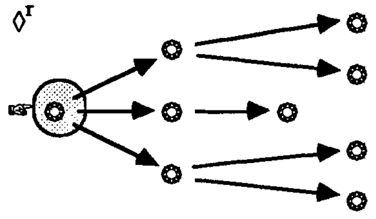




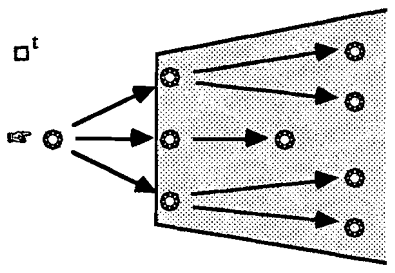
In the presence of a $\Box^{t}$-operator, $\Box^{\emptyset}$ may be interpreted as ‘all next’ and $\Diamond^{\emptyset}$ may be interpreted as ‘sometimes next’.

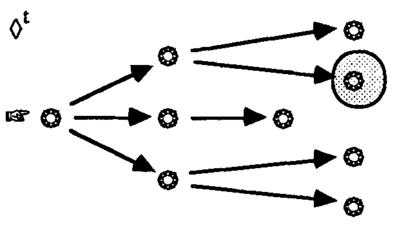
The operators $\Box^{r}$ (access to *all* directly accessible worlds including the actual world) and
$\Diamond^{r}$ (access to *some* directly accessible worlds including the actual world):





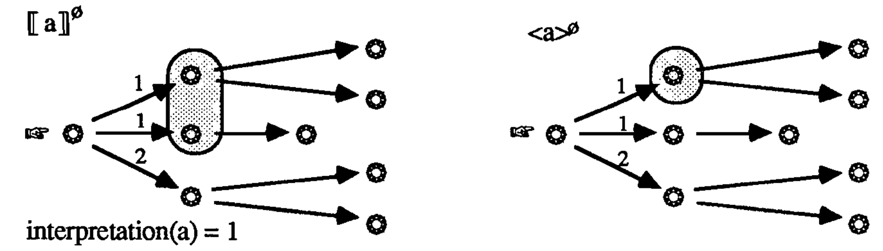
The operators $\Box^{t}$ (access to *all* directly and indirectly accessible worlds) and
$\Diamond^{t}$ (access to *some* directly and indirectly accessible worlds):





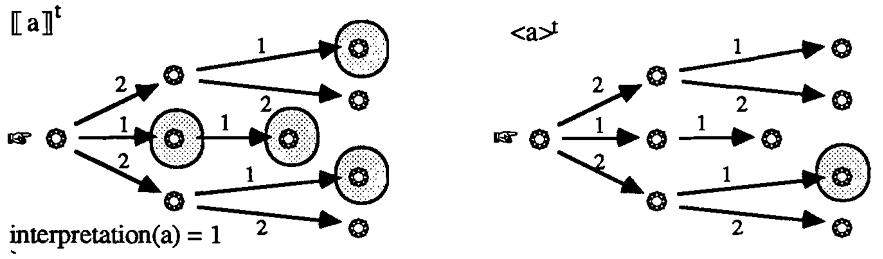
The sets of worlds that are accessed by $\Box^{rt}$ is the union of the corresponding sets for the basic operators.

The indexed operators $[\![a]\!]^{\emptyset}$ (access to *all* worlds which are directly accessed by a transition that is labeled with the value of 'a') and $<a>^{\emptyset}$ (access to *some* worlds which are directly accessed by a transition that is labeled with the value of 'a'):



In a process interpretation of the possible worlds structure (see above) a formula $[\![a]\!]^{\emptyset}\mathcal{F}$ may be interpreted: $\mathcal{F}$ holds next after process 'a' has performed an action.

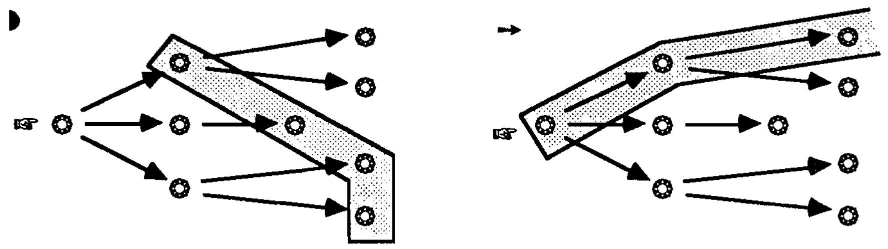
The indexed operators $[\![a]\!]^{t}$ (access to *all* worlds after an *a*-labeled transition where *a* is the interpretation of a. Only the last labels matter) and $<a>^{\emptyset}$ (access to *some* world after an *a*-labeled transition):



In a process interpretation of the possible worlds structure a formula $[\![a]\!]^{t}\mathcal{F}$ may be interpreted: $\mathcal{F}$ holds always after process 'a' has performed an action.

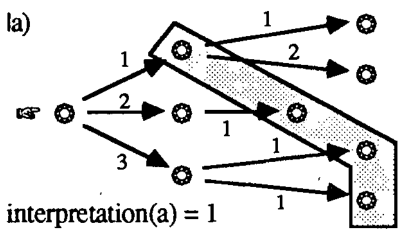
The remaining indexed operators $[\![a]\!]^{\mathcal{R}}$ and $<a>^{\mathcal{R}}$ work analogously but include the actual world.

The 'eventually' operator ◗ (access to a world on each path starting from the actual world) and the possibly henceforth operator ⤑ (access to a world on a particular path starting from the actual world):



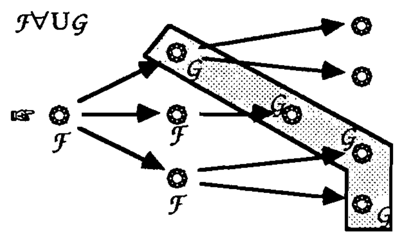
In the process interpretation the ◗-operator allows to express liveness properties like termination, deadlock freeness etc. ◗$\mathcal{F}$ says that regardless which path in the nondeterministic computation tree is followed, $\mathcal{F}$ will eventually hold.

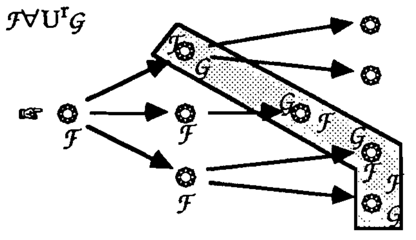
The indexed 'eventually' operator |a) (access to the *first* world after a transition labeled with the value of 'a' on each path starting from the actual world):



In the process interpretation a formula |a)$\mathcal{F}$ expresses: regardless how the nondeterminism in the computation tree is solved, process 'a' will eventually perform an action (fairness) and after that action, $\mathcal{F}$ will hold.

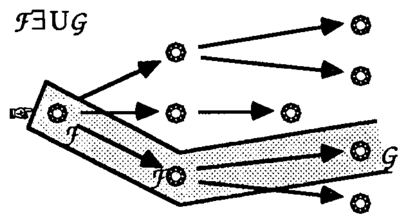
The ‘until’ operators $\forall U$ and $\forall U^r$:
$\mathcal{F}\forall U\mathcal{G}$ holds in the actual world if $\mathcal{G}$ holds eventually (◗$\mathcal{G}$) and $\mathcal{F}$ holds in all worlds before.
$\mathcal{F}\forall U^r\mathcal{G}$ holds in the actual world if $\mathcal{F}$ and $\mathcal{G}$ hold eventually (◗$(\mathcal{F}\wedge\mathcal{G})$) and $\mathcal{F}$ holds in all worlds before.

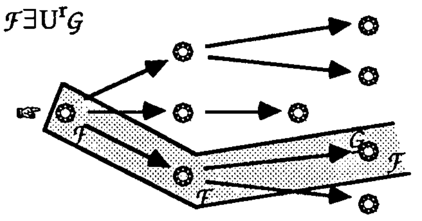




The ‘until’ operators allow to refer to limited areas in the possible worlds structure. In the process interpretation, ‘until’ operators are useful for expressing invariants which hold until a certain exception condition comes true.

The ‘until’ operators $\exists U$ and $\exists U^r$:
$\mathcal{F}\exists U\mathcal{G}$ holds in the actual world if $\mathcal{G}$ holds possibly ($\Diamond^{rt}\mathcal{G}$) and $\mathcal{F}$ holds in all worlds before.
$\mathcal{F}\exists U^r\mathcal{G}$ holds in the actual world if $\mathcal{F}$ and $\mathcal{G}$ hold possibly ($\Diamond^{rt}(\mathcal{F}\wedge\mathcal{G})$) and $\mathcal{F}$ holds in all worlds before.





We shall see that Context Logic allows to map all these complicated operators to a few basic concepts.

5.1 Syntax and Semantics

We define the syntax of our multi modal logic as an extension of the order-sorted predicate logic syntax, that means terms and atoms look like OSPL-terms and atoms, but formulae are composed using in addition the above set of operators.

Definition 5.1.1 (The Signature of Multi Modal Logic)
The signature definition for multi modal logic is exactly like the signature definition for OSPL. We assume that in the sort hierarchy there is a unique top sort D (for Domain). In the sequel ‘D’ always means this sort. ■

Definition 5.1.2 (Terms, Atoms, Literals and Formulae)
Terms, atoms, literals and formulae (“MM-formulae”) are built like OSPL-formulae.
The additional rules involving the modal operators are:

For $\mathcal{R}\in\{\emptyset, r, t, rt\}$: Whenever t is a term and $\mathcal{F}$ and $\mathcal{G}$ are MM-formulae, so are
$[\![t]\!]^{\mathcal{R}}\mathcal{F}$, $\langle t\rangle^{\mathcal{R}}\mathcal{F}$, ◗$\mathcal{F}$, $\twoheadrightarrow\mathcal{F}$, $|t)\mathcal{F}$, $\mathcal{F}\forall U\mathcal{G}$ and $\mathcal{F}\forall U^r\mathcal{G}$, $\mathcal{F}\exists U\mathcal{G}$ and $\mathcal{F}\exists U^r\mathcal{G}$. ■

We define the semantics of multi modal logic following the scheme of def. 2.1, i.e. we first define a “frame” as the kernel of the signature interpretation. Frames for MM-Logic are actually the usual possible worlds structures, however with labeled transitions and with Σ-structures as “worlds”.

Definition 5.1.3 **(M-Frames and M-Interpretations)**

By an *M-frame* $\mathfrak{F}_M$ for the signature Σ we understand any tuple $(\mathcal{W}, \mathfrak{R})$ where

1. $\mathcal{W}$ is a nonempty enumerable set of Σ-structures or "*worlds*".

   The domains are identical in all these Σ-structures (constant-domain assumption) and not empty.

2. $\mathfrak{R} := \{\mathfrak{R}^\emptyset, \mathfrak{R}^r, \mathfrak{R}^t, \mathfrak{R}^{rt}\}$ is a set of *serial* binary relations over $\mathcal{W}\times\mathcal{W}$ (seriality assumption).

   a) $\mathfrak{R}^\emptyset$ is the **basic discrete accessibility relation.**

   b) $\mathfrak{R}^r$ is the reflexive closure of $\mathfrak{R}^\emptyset$.

   c) $\mathfrak{R}^t$ is the transitive closure of $\mathfrak{R}^\emptyset$.

   d) $\mathfrak{R}^{rt}$ is the reflexive closure of $\mathfrak{R}^t$, i.e. it is the reflexive transitive closure of $\mathfrak{R}^\emptyset$.

   For a given world $\mathcal{W} \in \mathcal{W}$ let $\mathfrak{P}(\mathcal{W})$ denote the set of all **paths** starting with $\mathcal{W}$.

   Each path $\mathcal{P} \in \mathfrak{P}(\mathcal{W})$ is a maximal set of worlds

   i) where $\mathfrak{R}^{rt}$ is a total ordering on $\mathcal{P}$ (a $\leq$-ordering)

   ii) with $\mathcal{W}$ as smallest element.

   In the sequel a label ℓ is just a domain element.

   e) From each world $\mathcal{W}$ there are for each label ℓ $\mathfrak{R}^{\mathcal{X}}$-transitions associated with ℓ (seriality of labeled transitions).

   For transitive transitions which can be decomposed into an $\mathfrak{R}^{rt}$-transition followed by an $\mathfrak{R}^\emptyset$-transition only the label of the last $\mathfrak{R}^\emptyset$-transition matters, i.e. the $\mathfrak{R}^t$-transition is ℓ-labeled iff this last $\mathfrak{R}^\emptyset$-transition is ℓ-labeled.The same holds for an $\mathfrak{R}^{rt}$-transition which is not the identity.

   $\mathfrak{R}^{\mathcal{X}}(\ell)$ denotes the subrelation of the ℓ-labeled $\mathfrak{R}^{\mathcal{X}}$-transitions.

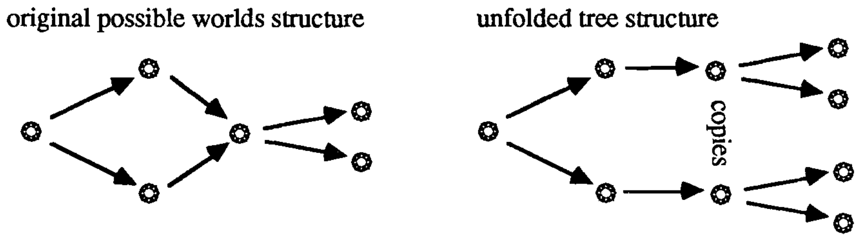
   f) The reflexive transitions are labeled with all possible labels.

   g) For each world $\mathcal{W}$, for each path $\mathcal{P} \in \mathfrak{P}(\mathcal{W})$ and for each label ℓ there is somewhere on $\mathcal{P}$ an ℓ-labeled $\mathfrak{R}^\emptyset$-transition (fairness assumption).

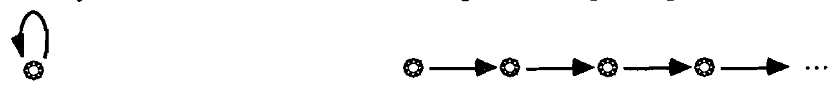
By a **signature interpretation** $\mathfrak{I}$ for the signature Σ understand any triple $(\mathfrak{F}_M, \mathcal{V}, \mathcal{W})$ where

- ➤ $\mathfrak{F}_M = (\mathcal{W}, \mathfrak{R})$ is an M-frame.
- ➤ $\mathcal{V}$ is a variable assignment, a Σ-assignment.
- ➤ $\mathcal{W}$ is an element of $\mathcal{W}$ (the *actual* world). ■

Remark: W.l.o.g we can assume that the $\mathfrak{R}^\emptyset$-relation is tree like. If it is not, the possible worlds structure can always be unfolded as in the examples below to make it a tree.



In case $\mathfrak{R}^\emptyset$ is already reflexive in a world, an infinite sequence of copies is generated:



■

Definition 5.1.4 **(Interpretation of Terms)**

Let $\mathfrak{I} = (\mathfrak{F}_M, \mathcal{V}, \mathcal{W})$ be a signature interpretation for the signature Σ.

$\mathfrak{I}$ can be turned into an homomorphism that evaluates terms in the actual world $\mathcal{W}$ by defining:

$\mathfrak{I}(x)$	$:= \mathcal{V}(x)$	if x is a variable symbol
$\mathfrak{I}(f(t_1,\ldots,t_n)$	$:= f_{\mathcal{W}}(\mathfrak{I}(t_1),\ldots,\mathfrak{I}(t_n))$	otherwise.

■

Some Notational Conventions:

- $\Im[x/\chi]$ denotes the interpretation which is like $\Im$ except that the variable assignment maps x to χ (i.e. it is like $\Im[x/\chi]_{\mathcal{V}}$ in chapter 4).
- $\Im[\mathcal{W}]$ denotes the interpretation which is like $\Im$ except that the actual world is $\mathcal{W}$ (i.e. it is like $\Im[W/\mathcal{W}]_C$ in chapter 4 where W denotes the context sort "worlds"). ■

Definition 5.1.5 **(The Satisfiability Relation)**

The satisfiability relation $\Vdash_M$ between signature interpretations $\Im = (\mathfrak{F}_M, \mathcal{V}, \mathcal{W}) = ((\boldsymbol{\mathcal{W}}, \Re), \mathcal{V}, \mathcal{W})$ and MM-formulae is defined as follows:

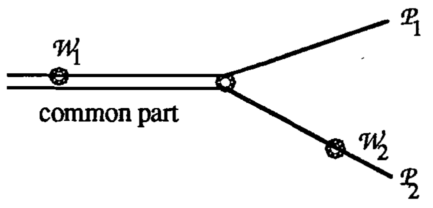
The predicate logic connectives $\wedge, \vee, \neg, \Rightarrow$ and $\Leftrightarrow$ are interpreted in the usual way.

Let $\mathcal{R} \in \{\emptyset, r, t, rt\}$

$\Im \Vdash_M P(t_1,\ldots,t_n)$	where P is a predicate symbol and the t_i are terms iff $(\Im(t_1),\ldots,\Im(t_n)) \in P_{\mathcal{W}}$
$\Im \Vdash_M \forall x{:}D\ \mathcal{F}$	iff for every $\chi \in D_{\mathcal{W}}$: $\Im[x/\chi] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \exists x\ \mathcal{F}$	iff for some $\chi \in D_{\mathcal{W}}$ with $\Im[x/\chi] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \Box^{\mathcal{R}} \mathcal{F}$	iff for every $\mathcal{W}_1 \in \boldsymbol{\mathcal{W}}$ with $\Re^{\mathcal{R}}(\mathcal{W}, \mathcal{W}_1)$: $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \Diamond^{\mathcal{R}} \mathcal{F}$	iff there is a $\mathcal{W}_1 \in \boldsymbol{\mathcal{W}}$ with $\Re^{\mathcal{R}}(\mathcal{W}, \mathcal{W}_1)$ and $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M [\![t]\!]^{\mathcal{R}} \mathcal{F}$	iff for every $\mathcal{W}_1 \in \boldsymbol{\mathcal{W}}$ with $(\mathcal{W}, \mathcal{W}_1) \in \Re^{\mathcal{R}}(\Im(t))$: $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \langle t\rangle^{\mathcal{R}} \mathcal{F}$	iff there is a $\mathcal{W}_1 \in \boldsymbol{\mathcal{W}}$ with $(\mathcal{W}, \mathcal{W}_1) \in \Re^{\mathcal{R}}(\Im(t))$ and $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \blacktriangleright \mathcal{F}$	iff on every path $\mathcal{P} \in \boldsymbol{\mathcal{P}}(\mathcal{W})$ there is a $\mathcal{W}_1 \in \mathcal{P}$ with $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \rightarrow \mathcal{F}$	iff there is a path $\mathcal{P} \in \boldsymbol{\mathcal{P}}(\mathcal{W})$ and for every $\mathcal{W}_1 \in \mathcal{P}$. $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \|t)\mathcal{F}$	iff on every path $\mathcal{P} \in \boldsymbol{\mathcal{P}}(\mathcal{W})$, $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$ holds in the first world $\mathcal{W}_1 \in \mathcal{P}$ that is accessed via an $\Im(t)$-labeled $\Re^{\emptyset}$-transition.
$\Im \Vdash_M \mathcal{F} \forall U\ \mathcal{G}$	iff on every path $\mathcal{P} \in \boldsymbol{\mathcal{P}}(\mathcal{W})$ there is a $\mathcal{W}_2 \in \mathcal{P}$ with $\Im[\mathcal{W}_2] \Vdash_M \mathcal{G}$ and for every world $\mathcal{W}_1 \in \mathcal{P}$ with $\Re^{t}(\mathcal{W}_1, \mathcal{W}_2)$: $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \mathcal{F} \forall U^{r}\ \mathcal{G}$	iff on every path $\mathcal{P} \in \boldsymbol{\mathcal{P}}(\mathcal{W})$ there is a $\mathcal{W}_2 \in \mathcal{P}$ with $\Im[\mathcal{W}_2] \Vdash_M \mathcal{G}$ and for every world $\mathcal{W}_1 \in \mathcal{P}$ with $\Re^{rt}(\mathcal{W}_1, \mathcal{W}_2)$: $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \mathcal{F} \exists U\ \mathcal{G}$	iff there is a path $\mathcal{P} \in \boldsymbol{\mathcal{P}}(\mathcal{W})$ and there is a $\mathcal{W}_2 \in \mathcal{P}$ with $\Im[\mathcal{W}_2] \Vdash_M \mathcal{G}$ and for every world $\mathcal{W}_1 \in \mathcal{P}$ with $\Re^{t}(\mathcal{W}_1, \mathcal{W}_2)$: $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im \Vdash_M \mathcal{F} \exists U^{r}\ \mathcal{G}$	iff there is a path $\mathcal{P} \in \boldsymbol{\mathcal{P}}(\mathcal{W})$ and there is a $\mathcal{W}_2 \in \mathcal{P}$ with $\Im[\mathcal{W}_2] \Vdash_M \mathcal{G}$ and for every world $\mathcal{W}_1 \in \mathcal{P}$ with $\Re^{rt}(\mathcal{W}_1, \mathcal{W}_2)$: $\Im[\mathcal{W}_1] \Vdash_M \mathcal{F}$.
$\Im$ *satisfies* $\mathcal{F}$	iff $\Im \Vdash_M \mathcal{F}$ ($\mathfrak{F}_M$ satisfies $\mathcal{F}$ *in the world* $\mathcal{W}$)
$\mathfrak{F}_M$ *satisfies* $\mathcal{F}$	iff it satisfies $\mathcal{F}$ in every world ■

It is now easy to verify that the dual operators are really dual.

Remark The semantics of the 'eventually' and 'until' operators contains an ambiguity of the following kind: A quantification "on every path there is a world..." may denote different worlds on paths with common parts:



Since in this case $\mathcal{W}_1$ is also an element of $\mathcal{P}_2$, $\mathcal{W}_1$ may serve as the world "that exists on $\mathcal{P}_2$" as well. Therefore in the sequel we always assume that the world closest to the beginning of the path, i.e. $\mathcal{W}_1$ in the example, is meant.

Definition 5.1.6 MM-Logic

Following the definition scheme of def. 2.1 for logics we define the multi modal logic MM as the tuple (syntax, semantics) where syntax consists of

- the set of all OSPL-signatures with unique top sorts (def. 5.1.1)
- the formation rules for OSPL-terms (def. 3.1.2)
- the formation rules for MM-formulae (def. 5.1.2)

and semantics consists of

- the function that maps a signature Σ to the M-interpretations over Σ (def. 5.1.3).
- the function that turns an M-interpretation into a homomorphism for terms (def. 5.1.4).
- the satisfiability relation $\models_M$ (def. 5.1.5). ∎

Lemma 5.1.7 Correspondences Between Paths and Natural Numbers.

a) Each path $\mathcal{P} \in \mathfrak{P}(\mathcal{W})$ in an M-frame $(\mathcal{W}, \mathfrak{R})$ is isomorphic to the set of natural numbers.

b) Let $\mathcal{N}^{\emptyset} := \{1\}$, $\mathcal{N}^{r} := \{0,1\}$, $\mathcal{N}^{t} := \{n \mid n > 0\}$ and $\mathcal{N}^{rt} := \{n \mid n \geq 0\}$.

For each $\mathfrak{R}^{\mathcal{R}}$-transition $(\mathcal{W},\mathcal{W}')$ there is an $n \in \mathcal{N}^{\mathcal{R}}$ such that $\mathcal{W}'$ is reached with n $\mathfrak{R}^{\emptyset}$-transitions.

Proof: a) Since $\mathfrak{R}^{rt}$ is the reflexive transitive closure of $\mathfrak{R}^{\emptyset}$ and since paths are ordered by $\mathfrak{R}^{rt}$, for each $\mathcal{W} \neq \mathcal{W}' \in \mathcal{P}$ there is a number n such that $\mathcal{W}_n = \mathcal{W}'$ and $\mathfrak{R}^{rt}(\mathcal{W},\mathcal{W}') \Leftrightarrow \mathfrak{R}^{\emptyset}(\mathcal{W},\mathcal{W}_1) \wedge \ldots \wedge \mathfrak{R}^{\emptyset}(\mathcal{W}_{n-1},\mathcal{W}_n)$.
Since $\mathcal{P}$ is a maximal set, all these $\mathcal{W}_i$ are in $\mathcal{P}$ and the ordering is $\mathcal{W} < \mathcal{W}_1 < \ldots$.

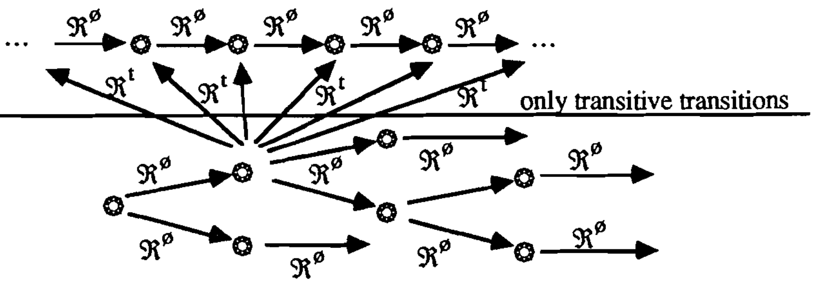
b) For each $\mathfrak{R}^{\mathcal{R}}$-transition $(\mathcal{W},\mathcal{W}')$ there is a path $\mathcal{P}$ crossing $\mathcal{W}$ and $\mathcal{W}'$. Using that paths are isomorphic to natural numbers, we obtain b) by a simple case analysis. ∎

In the definition of M-frames, def. 5.1.3, $\mathfrak{R}^t$ is defined as the transitive closure of $\mathfrak{R}^{\emptyset}$, and this is the reason why paths are isomorphic to natural numbers. Being the transitive closure is a property that cannot be expressed in first-order logic. Therefore we can expect difficulties with this strong condition on $\mathfrak{R}^t$. And in fact, the following example shows that in this case the compactness theorem does not hold for MM-Logic. Consider the following infinitely many formulae:

$$\Box^{\emptyset} P$$
$$\Box^{\emptyset}\Box^{\emptyset} P$$
$$\ldots$$
$$\Diamond^{t} \neg P$$

Each finite subset of this infinite set of formulae is satisfiable because the world denoted by the $\Diamond^t$-operator can be chosen far enough from the initial world. The whole set, however, is unsatisfiable since the sequence $\Box^{\emptyset}$, $\Box^{\emptyset}\Box^{\emptyset}$,... of possibility operators exhausts all worlds which are accessible from the initial world. There is no world left for the $\Diamond^t$-operator.

That means *the compactness theorem does not hold.* There are theorems whose proof requires infinitely many steps and therefore we cannot expect a complete calculus for the version of MM-Logic with the transitive closure interpretation of $\mathfrak{R}^t$. If we enforce the compactness property, we have to allow for nonstandard models where $\mathfrak{R}^t$ contains more than the transitive closure of $\mathfrak{R}^{\emptyset}$. Similar to nonstandard models in first-order arithmetic, in nonstandard MM-models there may be $\mathfrak{R}^t$-transitions into side chains which cannot be accessed by a sequence of $\mathfrak{R}^{\emptyset}$-transitions:



In order to show that the translation into CL we are going to define does not make things worse than they are already, we show a "weak completeness", i.e. each model of the translated MM-specifi- cation corresponds to a nonstandard model of the original MM-specification.

Since we cannot expect a complete proof theory for MM-Logic, the only thing we can do is to approximate the transitive closure of $\mathfrak{R}^t$ as far as possible and show that a complete calculus for the approximation is obtained. As in first-order arithmetic where standard natural numbers are approximated by a first-order induction scheme $P(0) \wedge (\forall n\ P(n) \Rightarrow P(n+1)) \Rightarrow \forall n\ P(n)$, we should incorporate a mechanism for proving induction axioms of the following kind:

If $\mathcal{F}$ holds in the initial world and for all worlds $\mathcal{W}$: $\mathcal{F}$ holds in $\mathcal{W}$ implies $\mathcal{F}$ holds in an $\mathfrak{R}^{\emptyset}$-successor world, then there is a path from the initial world where $\mathcal{F}$ holds everywhere.

or more formally: $\mathcal{F} \wedge \Box^{\pi}(\mathcal{F} \Rightarrow \Diamond^{\emptyset}\mathcal{F}) \Rightarrow \rightarrowtail\mathcal{F}$.

If the calculus can prove all formulae of this and similar kind we are yet not complete, but we can handle the theorems which are relevant for practical applications of MM-Logic, for example loop invariants in the process interpretation. Unfortunately, as we shall see in section 5.4, the problem of incorporating induction theorem proving in a calculus for MM-Logic is as complex as general induction theorem proving in predicate logic [Boyer&Moore 79]. Therefore this report does not consider induction theorem proving in more detail.

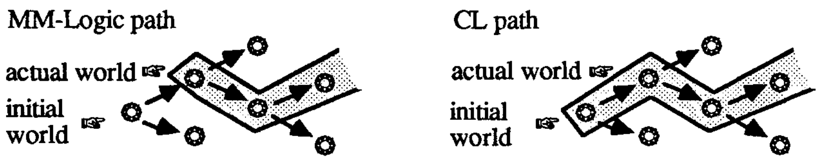
5.2 A Logic Morphism from Multi Modal Logic to Context Logic

We define a logic morphism Ψ (def. 2.3) from MM to CL. Its composition with the logic morphism Π from CL to OSPL enables proofs by translation into OSPL and refutation (prop. 2.4) with resolution and paramodulation.

Before we actually start defining the logic morphism we should introduce more or less informally the basic concepts of the CL-axiomatization of MM-Logic possible worlds structures. The main requirement for the CL-axiomatization is to replace the relational description of MM-Logic frames by a functional description where the argument-value relation of context access functions models the accessibility relations. Therefore we have to introduce functions mapping worlds to accessible worlds. But things are not so straightforward. In the semantics definition of the 'eventually' and 'until' operators there are transitions to worlds *on the current path*. Functions simulating these transitions need a world and a path as input. Therefore, besides the sort symbol W for world, we introduce a sort symbol WP as basic context sort denoting tuples <world, path> where the first component, the world-component is an element of the path-component. All context access functions now operate on these "WP-tuples".

In the sequel $X_{|W}$ denotes projection of the tuple X to its world component and $X_{|P}$ denotes projection to the path component.

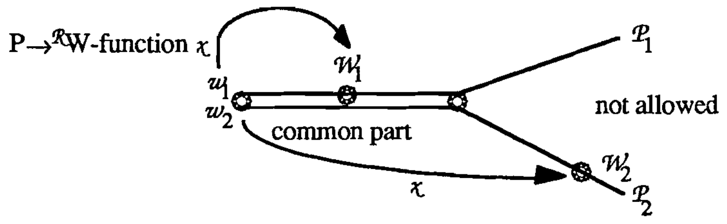
Before going into details of the context access functions, we should say a few words about the notion of paths in the CL-axiomatization of MM-Logic. (The actual axiomatization will be given a few pages below.) In MM-Logic we have defined $\mathfrak{P}(\mathcal{W})$ as the set of all paths *starting with the world* $\mathcal{W}$, i.e. a path starts with the actual world. This was necessary because the quantification "for all worlds on the actual path" in the semantics definition of some of the operators should only range over accessible worlds and not over worlds lying backwards to the actual world. In the CL-axiomatization we simplify the notion of a path a bit and define a path to start always at the initial world.



We don´t get into trouble with this definition because a quantification over all worlds on a path is replaced by a quantification over all context access functions that map the actual world to a world on the path, and these functions never move backwards in the possible worlds structure.

The unparametrized context access functions:

Each function x mapping a WP-tuple to another WP-tuple can be split into a composition of two primitive functions, MP(x) and MW(x), the first one changing only the path component and the second one moving the actual world along the changed path. (The other way round is not possible because you can´t move off the path. The analogy to linear algebra is therefore not very good.) As basic building blocks for the context access functions we introduce ‘W→P’-functions and ‘P→$^{\mathcal{R}}$W’-functions. The ‘W→P’-functions change only paths, and leave worlds untouched whereas the ‘P→$^{\mathcal{R}}$W’-functions move along the current path to $\mathfrak{R}^{\mathcal{R}}$-accessible worlds. We have to impose one restriction on the ‘P→$^{\mathcal{R}}$W’-functions which resolves the ambiguity mentioned in the remark after the semantics definition for the MM-Logic operators, def. 5.1.7. We do not allow a ‘P→$^{\mathcal{R}}$W’-function to access on two different paths with a common part two different worlds $\mathcal{W}_1$ and $\mathcal{W}_2$ when one of them lies on the common part, i.e. a situation like



never occurs. Therefore we get as a basic axiom that restricts the ‘P→$^{\mathcal{R}}$W’-functions:

$$\forall x,y,z{:}\text{‘P}\to^{\mathcal{R}}\text{W’}\ \forall p{:}\text{‘W}\to\text{P’}\ \ \forall w{:}WP\ (MP(x \circ z \circ p) \circ y)(w)_{|W} = x(w)_{|W} \Leftrightarrow x(w) = y(w)$$

which expresses that only the part of the path up to x(w) is relevant for x.

From the ‘W→P’-functions and ‘P→$^{\mathcal{R}}$W’-functions ‘W→$^{\mathcal{R}}$W’-functions are obtained as a composi- tion of a ‘W→P’-function and a ‘P→$^{\mathcal{R}}$W’-function. The axiomatization of these functions requires the introduction of the corresponding sort symbols together with the sort hierarchy and the specification of the sort declarations for the composition function ∘. To determine the sort hierarchy, we exploit that each $\mathfrak{R}^{\emptyset}$-transition is both a $\mathfrak{R}^{r}$-transition and a $\mathfrak{R}^{t}$-transition, and $\mathfrak{R}^{r}$-transitions as well as $\mathfrak{R}^{t}$-transitions are both $\mathfrak{R}^{rt}$-transitions. This is reflected in the following sort hierarchy:

$$\begin{tikzcd}
 & \text{‘P}\to^{rt}\text{W’} \arrow[dl, dash] \arrow[dr, dash] & & & & \text{‘W}\to^{rt}\text{W’} \arrow[dl, dash] \arrow[dr, dash] & \\
\text{‘P}\to^{r}\text{W’} \arrow[dr, dash] & & \text{‘P}\to^{t}\text{W’} \arrow[dl, dash] & & \text{‘W}\to^{r}\text{W’} \arrow[dr, dash] & & \text{‘W}\to^{t}\text{W’} \arrow[dl, dash] \\
 & \text{‘P}\to^{\emptyset}\text{W’} & & & & \text{‘W}\to^{\emptyset}\text{W’} &
\end{tikzcd}$$

Furthermore, since each ‘P→$^{\mathcal{R}}$W’-function is only a special version of ‘W→$^{\mathcal{R}}$W’-functions, we must add the relations ‘P→$^{\mathcal{R}}$W’ ⊑ ‘W→$^{\mathcal{R}}$W’. A ‘W→P’-function is a special ‘W→^{r}W’-function, therefore we add ‘P→W’ ⊑ ‘W→^{r}W’. Both ‘W→P’-functions and ‘P→$^{\mathcal{R}}$W’-functions contain the identity function. Therefore we finally add a sort ‘ID’ denoting the identity function only and obtain the complete sort hierarchy:

$$\begin{tikzcd}
 & & & \text{‘W}\to^{rt}\text{W’} \arrow[dll, dash] \arrow[d, dash] \arrow[drr, dash] & & & \\
 & \text{‘W}\to^{r}\text{W’} \arrow[dl, dash] \arrow[dr, dash] \arrow[drrr, dash] & & \text{‘P}\to^{rt}\text{W’} \arrow[dl, dash] \arrow[drrr, dash] & & \text{‘W}\to^{t}\text{W’} \arrow[dl, dash] \arrow[dr, dash] & \\
\text{‘W}\to\text{P’} \arrow[dr, dash] & & \text{‘P}\to^{r}\text{W’} \arrow[dl, dash] \arrow[drr, dash] & & \text{‘W}\to^{\emptyset}\text{W’} \arrow[d, dash] & & \text{‘P}\to^{t}\text{W’} \arrow[dll, dash] \\
 & \text{‘ID’} & & & \text{‘P}\to^{\emptyset}\text{W’} & &
\end{tikzcd}$$

The sort declarations for the composition function $\circ$ can be deduced straightforwardly from the intended meaning of the '$W \rightarrow {}^{\mathcal{R}}W$'-functions:

$\circ$: '$W \rightarrow {}^{p}W$' $\times$ '$W \rightarrow {}^{q}W$' $\rightarrow$ '$W \rightarrow {}^{s}W$' and $\circ$: '$W \rightarrow {}^{\mathcal{R}}W$' $\times$ '$W \rightarrow P$' $\rightarrow$ '$W \rightarrow {}^{\mathcal{R}}W$'

'$P \rightarrow {}^{p}W$' $\times$ '$P \rightarrow {}^{q}W$' $\rightarrow$ '$P \rightarrow {}^{s}W$' '$W \rightarrow P$' $\times$ '$W \rightarrow {}^{\mathcal{R}}W$' $\rightarrow$ '$W \rightarrow {}^{\mathcal{R}}W$'

'$W \rightarrow P$' $\times$ '$W \rightarrow P$' $\rightarrow$ '$W \rightarrow P$'

where s is derived from p and q with the following (symmetric and associative) matrix:

p \ q	ø	r	t	rt
ø	t	t	t	t
r	t	rt	t	rt
t	t	t	t	t
rt	t	rt	t	rt

Since '$W \rightarrow {}^{rt}W$' is the top sort in the sort hierarchy, in the sequel we usually write axioms with quantifications only over '$W \rightarrow {}^{rt}W$'-functions. Together with overloaded sort declarations for the functionals to be introduced below, we automatically get the right instances of the axioms for all other context access functions.

Each '$W \rightarrow {}^{\mathcal{R}}W$'-function can be decomposed into a composition of a '$W \rightarrow P$'-function and a '$P \rightarrow {}^{\mathcal{R}}W$'-function. To do the composition syntactically we introduce the already mentioned function symbols MP ("move path") and MW ("move world").

The axiomatization of MP and MW is:

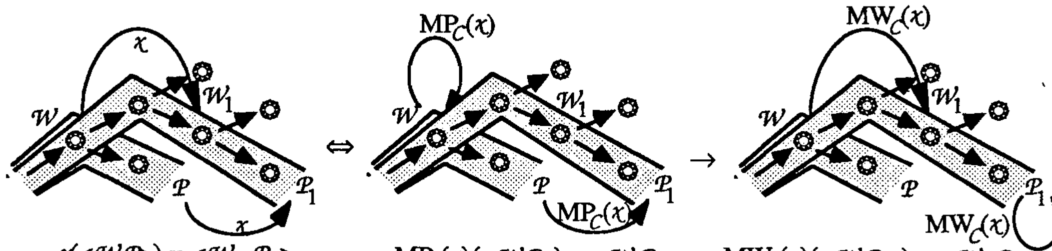
MP: '$W \rightarrow {}^{\mathcal{R}}W$' $\rightarrow$ '$W \rightarrow P$' and MW: '$W \rightarrow {}^{\mathcal{R}}W$' $\rightarrow$ '$P \rightarrow {}^{\mathcal{R}}W$'

$\forall x$:'$P \rightarrow {}^{rt}W$' $MW(x) = x$ $\forall x$:'$P \rightarrow {}^{rt}W$' $MP(x) = ID$ (= identity)

$\forall x$:'$W \rightarrow P$' $MW(x) = ID$ $\forall x$:'$W \rightarrow P$' $MP(x) = x$

$\forall x$:'$W \rightarrow {}^{rt}W$' $x = MP(x) \circ MW(x)$ $\forall x,y$:'$W \rightarrow {}^{rt}W$' $MW(x \circ y) = MW(x) \circ MW(y)$.

or graphically:



$x(<\mathcal{W},\mathcal{P}>) = <\mathcal{W}_1,\mathcal{P}_1>$ $MP_C(x)(<\mathcal{W},\mathcal{P}>) = <\mathcal{W},\mathcal{P}_1>$ $MW_C(x)(<\mathcal{W},\mathcal{P}_1>) = <\mathcal{W}_1,\mathcal{P}_1>$

The parametrized context access functions:

Since for a label ℓ we do not have from each world on each path an ℓ-labeled transition, but only on some paths ℓ-labeled transitions, we can't decompose the parametrized context access functions into ones which change only the path and others which move along a path, i.e. we can't have '$D,P \rightarrow {}^{\mathcal{R}}W$'-functions but only '$D,W \rightarrow {}^{\mathcal{R}}W$'-functions. Applied to a label ℓ they, however, produce a normal '$W \rightarrow {}^{\mathcal{R}}W$'-function. Consequently, the '$D,W \rightarrow {}^{\mathcal{R}}W$' part of the sort hierarchy looks exactly like the '$W \rightarrow {}^{\mathcal{R}}W$' part:

$$\begin{tikzcd}
 & \text{'}D,W \rightarrow {}^{rt}W\text{'} \arrow[dl, dash] \arrow[dr, dash] & \\
\text{'}D,W \rightarrow {}^{r}W\text{'} \arrow[dr, dash] & & \text{'}D,W \rightarrow {}^{t}W\text{'} \arrow[dl, dash] \\
 & \text{'}D,W \rightarrow {}^{\emptyset}W\text{'} &
\end{tikzcd}$$

and the corresponding sort declaration for the composition function

$\circ$:'$D,W \rightarrow {}^{p}W$' $\times$ '$D,W \rightarrow {}^{q}W$' $\rightarrow$ '$D,W \rightarrow {}^{s}W$'

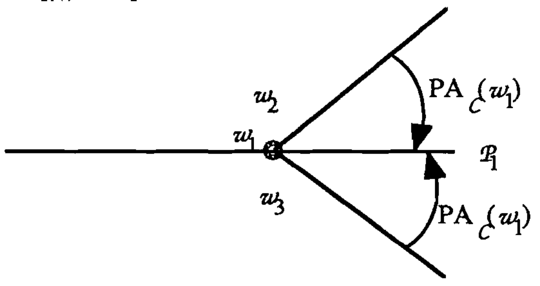
is analogue to the sort declarations for the '$W \rightarrow {}^{p}W$' sorts.

Since only the last label in a sequence of transitions matters a labeled transitive transition can be obtained by the composition of an arbitrary transition and a labeled transition. Therefore we introduce a special functional LT (labeled transtion) that composes a two such transitions to a labeld transition, i.e.

$$\forall x:`W\to^{rt}W', y:`D,W\to^{rt}W' \ \forall l:D\ (x \circ y(l)) = LT(x, y)(l)$$

The set of 'W→P'-functions is intended to describe transitions between all paths crossing a given world. That means in particular that for a given WP-tuple w_1 all other WP-tuples w_2 with the same world can be mapped to w_1, i.e. $\forall w_1:WP\ \exists p:`W\to P'\ \forall w_2:WP\ w_{1|W} = w_{2|W} \Rightarrow p(w_2) = w_1$

The Skolem function for p is a function PA: WP → 'W→P' that returns for a given WP-tuple a 'W→P'-function which maps all paths crossing $w_{1|W}$ to w_1.



So far we have introduced context access functions to model $\Re^{\mathcal{R}}$-transitions and we have motivated the functionals MP, MW and PA. But we have not introduced any means to represent the correlations between the different types of accessibility relations.

Basic transitions:

A transition in the basic accessibility relation $\Re^{\emptyset}$ on a particular path is just one step forward. Therefore we introduce a function symbol +1:'P→$^{\emptyset}$W' as the unique 'P→$^{\emptyset}$W'-function. Composed with 'W→P'-functions we obtain all context access functions related to a branching $\Re^{\emptyset}$-relation. '+1' corresponds to the successor function in the Péano axiomatization of natural numbers.

Reflexive versus nonreflexive transitions:

The correlation between the nonreflexive and the reflexive accessibility relations in terms of context access functions is that the reflexive functions operate on a world either as the identity or as a corresponding nonreflexive function. To express this syntactically we introduce a functional -R that "removes" the reflexive part from a context access function. The type declarations for -R are:

-R: 'W→^{rt}W' → 'W→^{t}W'	-R: 'P→^{rt}W' → 'P→^{t}W'
-R: 'W→^{r}W' → 'W→$^{\emptyset}$W'	-R: 'P→^{r}W' → 'P→$^{\emptyset}$W'

and the axiom describing -R is: $\forall x:`W\to^{rt}W'\ \forall w:WP\ \ x(w) = w \ \vee\ x(w) = -R(x)(w)$.

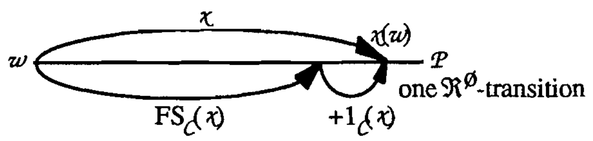
Transitive versus nontransitive transitions:

Since the transitive accessibility relation is simply the transitive closure of the basic one, the correlations between the basic and the transitive accessibility relation is that each transitive transition is either already a basic transition or it is decomposable into another transitive transition followed by a basic transition, i.e.

$$\forall x:`P\to^{t}W'\ \forall w:WP\ \ x(w) = +1(w) \vee x(w) = (FS(x) \circ +1)(w).$$

where FS: 'P→^{t}W' → 'P→^{t}W' (FS means 'first steps') and FS(x) makes one step shorter than x itself.

The interpretation of FS in a structure C is graphically:



Actually this axiom corresponds to one of the Péano axioms for natural numbers.

For reflexive-transitive transitions we can optimize the above correlations a bit: A reflexive-transitive transition either remains where it is or it can be decomposed into another reflexive-transitive transition followed by a basic transition. To express this we extend the meaning of FS, i.e. we introduce another sort declaration FS: $`P\rightarrow^{rt}W' \rightarrow `P\rightarrow^{rt}W'$ and add the axiom: $\forall x{:}`P\rightarrow^{rt}W'\ \forall w{:}WP\ \ x(w) = w \ \vee\ x(w) = (FS(x) \circ +1)(w)$.

Paths:

The remaining thing to be done is to axiomatize paths. There are two main conditions describing paths: A path is a totally ordered set, and the ordering is determined by $\Re^{rt}$-accessibility.

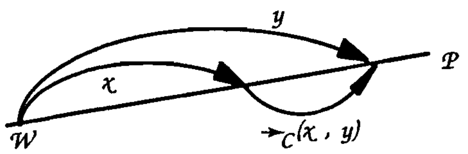
To axiomatize these conditions we introduce a relation symbol $\leq$: WP×WP and axiomatize it as a total ordering on path. The totality axiom is $\forall x,y{:}`P\rightarrow^{rt}W'\ \forall w{:}WP\ \ x(w) \leq y(w) \ \vee\ y(w) \leq x(w)$

i.e. only worlds on the same path are compared. The second condition actually consists of two parts. One part, $\Re^{rt}$-accessible worlds on a path are in the $\leq$-relation, is simply expressed by

$$\forall x,y{:}`P\rightarrow^{rt}W'\ \forall w_1,w_2{:}WP\ w_2 = x(w_1) \Rightarrow\ w_1 \leq w_2$$

The axiomatization of the second part, two worlds on a path being in the $\leq$-relation are $\Re^{rt}$-accessible, needs a new functional $\rightarrowtail$: $`P\rightarrow^{rt}W' \times `P\rightarrow^{rt}W' \rightarrow `P\rightarrow^{rt}W'$.

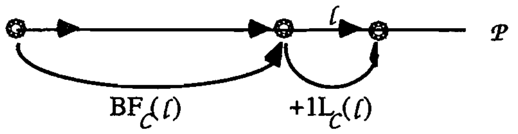
$\rightarrowtail$ denotes in a certain sense the difference between two worlds



and is axiomatized with $\forall x,y{:}`P\rightarrow^{rt}W'\ \forall w{:}WP\ \ x(w) \leq y(w) \Rightarrow (x \circ\!\rightarrowtail\!(x, y))(w) = y(w)$.

As an auxiliary predicate, a <-predicate with the usual meaning will also be introduced.

To express the semantics of the indexed 'eventually' operator ⟨...⟩, accessing on each path for a label l the world after the first l-labeled transition, we introduce a function BF(l) which takes a label l and returns a function that maps the current world to the world *before* the first l-labeled $\Re^{\emptyset}$-transition on the current path. The remaining l-labeled $\Re^{\emptyset}$-transition is described by a function +1L(l):



BF is described with the following axiom which expresses both, that the transition following BF(l)(w) is l-labeled and that this transition is the first one of this kind on the path.

$$\forall x{:}`P\rightarrow^{rt}W'\ \forall y{:}`D,W\rightarrow^{\emptyset}W'\ \forall l{:}D\ \forall w{:}WP\ (x \circ y(l))(w) = (x \circ +1)(w) \ \Rightarrow\ BF(l)(w) \leq x(w).$$

We are now going to turn the informal description of the CL-axiomatization of MM-Logic possible worlds structures into a formal definition for the logic morphism Ψ.

Definition 5.2.1 The Signature Morphism Ψ_Σ.

A MM-signature Σ is mapped to a CL-signature as follows:

1. The MM-signature becomes the domain part of the CL-signature.
2. The context part of the CL-signature is created from scratch:
   a) It is a functional signature (def. 3.4.1) over the basic context sort WP (for tuples <world,path>) and with interpretation context sort W (for worlds).

b) The sort lattice for the functional sorts is

$$\begin{tikzcd}
& & \text{‘W}\to^{rt}\text{W’} \arrow[dl, dash] \arrow[d, dash] \arrow[dr, dash] & \\
& \text{‘W}\to^{r}\text{W’} \arrow[dl, dash] \arrow[d, dash] \arrow[dr, dash] & \text{‘P}\to^{rt}\text{W’} \arrow[dl, dash] \arrow[dr, dash] & \text{‘W}\to^{t}\text{W’} \arrow[dl, dash] \arrow[d, dash] \\
\text{‘W}\to\text{P’} \arrow[d, dash] & \text{‘P}\to^{r}\text{W’} \arrow[dl, dash] \arrow[dr, dash] & \text{‘W}\to^{\emptyset}\text{W’} \arrow[d, dash] & \text{‘P}\to^{t}\text{W’} \arrow[dl, dash] \\
\text{‘ID’} & & \text{‘P}\to^{\emptyset}\text{W’} &
\end{tikzcd}$$

$$\begin{tikzcd}
& \text{‘D,W}\to^{rt}\text{W’} \arrow[dl, dash] \arrow[dr, dash] & \\
\text{‘D,W}\to^{r}\text{W’} \arrow[dr, dash] & & \text{‘D,W}\to^{t}\text{W’} \arrow[dl, dash] \\
& \text{‘D,W}\to^{\emptyset}\text{W’} &
\end{tikzcd}$$

c) The sort declarations for the composition function ∘ are:

∘: ‘W→^{p}W’ × ‘W→^{q}W’ → ‘W→^{s}W’ and ∘: ‘W→$^{\mathcal{R}}$W’ × ‘W→P’ → ‘W→$^{\mathcal{R}}$W’
‘P→^{p}W’ × ‘P→^{q}W’ → ‘P→^{s}W’ ‘W→P’ × ‘W→$^{\mathcal{R}}$W’ → ‘W→$^{\mathcal{R}}$W’
‘D,W→^{p}W’ × ‘D,W→^{q}W’ → ‘D,W→^{s}W’ ‘W→P’ × ‘W→P’ → ‘W→P’

where s is derived from p and q with the following matrix:

p \ q	∅	r	t	rt
∅	t	t	t	t
r	t	rt	t	rt
t	t	t	t	t
rt	t	rt	t	rt

The sort declarations for the application function ↓ are:

↓:	‘X→$^{\mathcal{R}}$Y’ × WP → WP	for X,Y ∈ {W, P}
	‘D,W→$^{\mathcal{R}}$W’ × D → ‘W→$^{\mathcal{R}}$W’	for the top domain sort D.

d) The following additional constant, function and predicate symbols are added:

Constant symbols:

ID:	‘ID’	(Two identity functions)
IDL	‘D,W→^{r}W’	
+1:	‘P→$^{\emptyset}$W’	
+1L:	‘D,W→$^{\emptyset}$W’	

Function symbols:

PW:	WP → W	
PA:	WP → ‘W→P’	
BF:	D → ‘P→^{rt}W’	
MP:	‘W→^{rt}W’ → ‘W→P’	
MW:	‘W→$^{\mathcal{R}}$W’ → ‘P→$^{\mathcal{R}}$W’	
LT:	‘W→^{s}W’ × ‘D,W→^{q}W’ → ‘D,W→^{s}W’,	
	s is derived from p and q according to the above matrix.	
-R:	‘W→^{rt}W’ → ‘W→^{t}W’	‘P→^{rt}W’ → ‘P→^{t}W’
	‘W→^{r}W’ → ‘W→$^{\emptyset}$W’	‘P→^{r}W’ → ‘P→$^{\emptyset}$W’
FS:	‘P→^{t}W’ → ‘P→^{t}W’	
	‘P→^{rt}W’ → ‘P→^{rt}W’	
➛:	‘P→^{rt}W’ × ‘P→^{rt}W’ → ‘P→^{rt}W’	

Predicate symbols:

≤:	WP×WP
<:	WP×WP (≤ and ≠)

f) The symbol variation function $\mathcal{SV}$ maps all domain function symbols to PW. ■

Definition 5.2.2 The Formula Morphism $\Psi_{\mathcal{F}}$.

$\Psi_{\mathcal{F}}$ leaves MM-terms unchanged and maps MM-formulae to domain formulae in CL.

Corresponding to the inductive definition of MM-formulae, the translation rules are:

Formulae with the predicate logic junctors and quantifiers as top operators are translated by leaving their structure unchanged and just translating their subformulae.

The translation rules for the modal operators are:

$\Psi_{\mathcal{F}}(\Box^{\mathcal{R}}\mathcal{F}) = \forall x{:}\text{‘}W\rightarrow^{\mathcal{R}}W\text{’}\ \Psi_{\mathcal{F}}(\mathcal{F})\ (\Leftrightarrow \forall p{:}\text{‘}W\rightarrow P\text{’}\ \forall x{:}\text{‘}P\rightarrow^{\mathcal{R}}W\text{’}\ \Psi_{\mathcal{F}}(\mathcal{F}))$

$\Psi_{\mathcal{F}}(\Diamond^{\mathcal{R}}\mathcal{F}) = \exists x{:}\text{‘}W\rightarrow^{\mathcal{R}}W\text{’}\ \Psi_{\mathcal{F}}(\mathcal{F})\qquad (\Leftrightarrow \exists p{:}\text{‘}W\rightarrow P\text{’}\ \exists x{:}\text{‘}P\rightarrow^{\mathcal{R}}W\text{’}\ \Psi_{\mathcal{F}}(\mathcal{F}))$

$\Psi_{\mathcal{F}}([\![t]\!]^{\mathcal{R}}\mathcal{F}) = \forall{\downarrow}(x{:}\text{‘}D,W\rightarrow^{\mathcal{R}}W\text{’}, z{:}S(t)=t)\ \Psi_{\mathcal{F}}(\mathcal{F})$ (S(t) denotes the sort of t)

$\Psi_{\mathcal{F}}(\langle t\rangle^{\mathcal{R}}\mathcal{F}) = \exists{\downarrow}(x{:}\text{‘}D,W\rightarrow^{\mathcal{R}}W\text{’}, z{:}S(t)=t)\ \Psi_{\mathcal{F}}(\mathcal{F})$

$\Psi_{\mathcal{F}}(\text{◗}\mathcal{F}) = \forall p{:}\text{‘}W\rightarrow P\text{’}\ \exists x{:}\text{‘}P\rightarrow^{rt}W\text{’}\ \Psi_{\mathcal{F}}(\mathcal{F})$

$\Psi_{\mathcal{F}}(\rightarrow\mathcal{F}) = \exists p{:}\text{‘}W\rightarrow P\text{’}\ \forall x{:}\text{‘}P\rightarrow^{rt}W\text{’}\ \Psi_{\mathcal{F}}(\mathcal{F})$

$\Psi_{\mathcal{F}}(|t)\mathcal{F}) = \forall p{:}\text{‘}W\rightarrow P\text{’}\ \wp(BF(t)\circ +1)\ \Psi_{\mathcal{F}}(\mathcal{F})$

$\Psi_{\mathcal{F}}(\mathcal{F}\,\forall U\ \mathcal{G}) = \forall p{:}\text{‘}W\rightarrow P\text{’}\ \exists x{:}\text{‘}P\rightarrow^{rt}W\text{’}\ (\Psi_{\mathcal{F}}(\mathcal{G}) \wedge \forall y{:}\text{‘}P\rightarrow^{rt}W\text{’-}x\ ({<}(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}))).$

$\Psi_{\mathcal{F}}(\mathcal{F}\,\forall U^{r}\ \mathcal{G}) = \forall p{:}\text{‘}W\rightarrow P\text{’}\ \exists x{:}\text{‘}P\rightarrow^{rt}W\text{’}\ (\Psi_{\mathcal{F}}(\mathcal{G}) \wedge \forall y{:}\text{‘}P\rightarrow^{rt}W\text{’-}x\ ({\leq}(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}))).$

$\Psi_{\mathcal{F}}(\mathcal{F}\,\exists U\ \mathcal{G}) = \exists p{:}\text{‘}W\rightarrow P\text{’}\ \exists x{:}\text{‘}P\rightarrow^{rt}W\text{’}\ (\Psi_{\mathcal{F}}(\mathcal{G}) \wedge \forall y{:}\text{‘}P\rightarrow^{rt}W\text{’-}x\ ({<}(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}))).$

$\Psi_{\mathcal{F}}(\mathcal{F}\,\exists U^{r}\ \mathcal{G}) = \exists p{:}\text{‘}W\rightarrow P\text{’}\ \exists x{:}\text{‘}P\rightarrow^{rt}W\text{’}\ (\Psi_{\mathcal{F}}(\mathcal{G}) \wedge \forall y{:}\text{‘}P\rightarrow^{rt}W\text{’-}x\ ({\leq}(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}))).$

Of course the introduced variables have to be completely new ones. ■

Definition 5.2.3 The Specification Morphism $\Psi_{\mathcal{S}}$

The specification morphism $\Psi_{\mathcal{S}}$ uses Ψ_{Σ} for translating MM-signatures into CL-signatures and $\Psi_{\mathcal{F}}$ for translating MM-formulae into CL-formulae. Furthermore, it adds the necessary axioms for the application function $\downarrow$ and the composition function $\circ$ (def. 3.4.1,5) to make the context part of the CL-specification a functional specification. And finally it adds the axioms which characterize possible worlds structures in terms of accessibility functions. (We use a second order syntax to make the axioms more readable. The first-order version of terms like x(y) is $\downarrow(x, y)$).

Characterization of $\circ$ and $\downarrow$:

A1 $\forall x,y{:}\text{‘}W\rightarrow^{rt}W\text{’}\ \forall w{:}WP\ \ {\downarrow}(x, w) = {\downarrow}(y, w) \Rightarrow x = y$

A2 $\forall x,y{:}\text{‘}D,W\rightarrow^{rt}W\text{’}\ \forall l{:}D\ \forall w{:}WP\ \ {\downarrow}({\downarrow}(x, l), w) = {\downarrow}({\downarrow}(y, l), w) \Rightarrow x = y$

A3 $\forall x,y{:}\text{‘}W\rightarrow^{rt}W\text{’}\ \forall w{:}WP\ \ {\downarrow}(x \circ y, w) = {\downarrow}(y, {\downarrow}(x, w))$

A4 $\forall x,y{:}\text{‘}D,W\rightarrow^{rt}W\text{’}\ \forall l{:}D\ \ {\downarrow}(x \circ y, l) = {\downarrow}(x, l) \circ {\downarrow}(y, l)$

Identity functions.

B1 $\forall w{:}WP\ \ {\downarrow}(ID, w) = w$

B2 $\forall w{:}WP\ \forall l{:}D\ \ {\downarrow}({\downarrow}(IDL, l), w) = w$ (The reflexive transitions are labeled with all labels.)

Characterization of the ‘W→P’-functions

C1 $\forall p{:}\text{‘}W\rightarrow P\text{’}\quad MP(p) = p$

C2 $\forall p{:}\text{‘}W\rightarrow P\text{’}\quad MW(p) = ID$

C3 $\forall\ w_1,w_2{:}WP\quad PW(w_1) = PW(w_2) \Rightarrow w_2 = PA(w_2)(w_1)$

Characterization of the ‘P→$^{\mathcal{R}}$W’-functions.

D1 $\forall x{:}\text{‘}P\rightarrow^{rt}W\text{’}\quad MW(x) = x$

D2 $\forall x{:}\text{‘}P\rightarrow^{rt}W\text{’}\quad MP(x) = ID$

D3 $\forall x,y,z{:}\text{‘}P\rightarrow^{rt}W\text{’}\ \forall p{:}\text{‘}W\rightarrow P\text{’}\ \forall w{:}WP\ \ x(w) = y(w) \Rightarrow PW(MP(x \circ z \circ p) \circ y)(w)) = PW(x(w))$

D4 $\forall x,y,z{:}\text{‘}P\rightarrow^{rt}W\text{’}\ \forall p{:}\text{‘}W\rightarrow P\text{’}\ \forall w{:}WP\ \ x(w) = y(w) \Leftarrow PW(MP(x \circ z \circ p) \circ y)(w)) = PW(x(w))$

D5 $\forall w_1,w_2{:}WP\quad +1(w_1) = +1(w_2) \Rightarrow w_1 = w_2$ (injectivity)

Characterization of the 'W$\to^{\mathcal{R}}$W'-functions.

E1	$\forall x,y$:'W$\to^{rt}$W'	$x = MP(x) \circ MW(x)$
E2	$\forall x,y$:'W$\to^{rt}$W' $\forall w$:WP	$x(w) = y(w) \Rightarrow MP(x)(w) = MP(y)(w)$
E3	$\forall x,y$:'W$\to^{rt}$W' $\forall w$:WP	$x(w) = y(w) \Rightarrow MW(x)(w) = MW(y)(w)$
E4	$\forall x,y$:'W$\to^{rt}$W'	$MW(x \circ y) = MW(x) \circ MW(y)$

Relations between the different transitions:

F1	$\forall x$:'P$\to^{\emptyset}$W'	$x = +1$	(basic transitions)
F2	$\forall x$:'W$\to^{rt}$W' $\forall w$	$x(w) = w \vee x(w) = -R(x)(w)$	(reflexive transitions)
F3	$\forall x$:'W$\to^{t}$W' $\forall w$:WP	$x(w) \neq w$	
F4	$\forall x$:'P$\to^{t}$W' $\forall w$:WP	$x(w) = +1(w) \vee x(w) = (FS(x) \circ +1)(w)$	(transitive transitions)
F5	$\forall x$:'P$\to^{rt}$W' $\forall w$:WP	$x(w) = w \vee x(w) = (FS(x) \circ +1)(w)$.	(reflexive transitive transitions)

$\leq$ is a total ordering on paths.

G1	$\forall w$:WP	$w \leq w$	(reflexivity)
G2	$\forall w_1,w_2,w_3$:WP	$w_1 \leq w_2 \wedge w_2 \leq w_3 \Rightarrow w_1 \leq w_3$	(transitivity)
G3	$\forall w_1,w_2$:WP	$w_1 \leq w_2 \wedge w_2 \leq w_1 \Rightarrow w_1 = w_2$	(antisymmetry)
G4	$\forall x,y$:'P$\to^{rt}$W' $\forall w$:WP	$x(w) \leq y(w) \vee y(w) \leq x(w)$.	(totality on paths)

Definition of <.

H1 $\forall w_1,w_2$:WP $w_1 < w_2 \Rightarrow w_1 \leq w_2$

H2 $\forall w_1,w_2$:WP $w_1 < w_2 \Rightarrow w_1 \neq w_2$

H3 $\forall w_1,w_2$:WP $w_1 < w_2 \Leftarrow w_1 \leq w_2 \wedge w_1 \neq w_2$.

Characterization of paths

I1 $\forall x$:'P$\to^{rt}$W' $\forall w_1,w_2$:WP $w_2 = x(w_1) \Rightarrow w_1 \leq w_2$

I2 $\forall x,y$:'P$\to^{rt}$W' $\forall w$:WP $x(w) \leq y(w) \Rightarrow (x \circ\!\!\rightarrow(x, y))(w) = y(w)$.

Labeled transitions on paths

J1 $\forall x$:'P$\to^{rt}$W' $\forall y$:'D,W$\to^{\emptyset}$W' $\forall l$:D $\forall w$:WP $(x \circ y(l))(w) = (x \circ +1)(w) \Rightarrow BF(l)(w) \leq x(w)$.

J2 $\forall l$:D $BF(l) \circ +1L(l) = BF(l) \circ +1$

J3 $\forall x$:'P$\to^{rt}$W' $\forall y$:'D,W$\to^{\emptyset}$W' $\forall l$:D $x \circ y(l) = LT(x,y)(l)$ ■

In the subsequent sections of this chapter the soundness and completeness of the specification morphism will be investigated.

Lemma 5.2.4 Well Formedness of $\Psi_{\mathcal{S}}(\mathcal{S})$

If $\mathcal{S} = (\Sigma, \mathcal{F})$ is a correct MM-specification then $\Psi_{\mathcal{S}}(\mathcal{S})$ is a syntactically correct CL-specification.

Proof: We have to check

a) whether $\Psi_{\mathcal{S}}(\mathcal{S})$ is a functional OSPL-specification according to def. 3.4.1 and 4.1.1,

b) whether the translated MM-formulae are well formed CL domain $\Psi_{\Sigma}(\Sigma)$-formulae and

c) whether the axioms generated by $\Psi_{\mathcal{S}}(\mathcal{S})$ are well formed CL context $\Psi_{\Sigma}(\Sigma)$-formulae.

a) 1. The functional sorts are 'WP$\to^{q}$WP' and 'D,WP$\to^{q}$WP'. We use 'W$\to$P' etc. only as abbreviations. Therefore they meet the conditions in 3.4.1,1 and 4.1.1,c.

2. Since there is only one variant of 'D,WP$\to$WP' sorts where D is the unique top domain sort, the subsort declarations meet the condition 3.4.1,2.

3. The sort declarations for $\downarrow$ have the structure:

   $\downarrow$: ‘$WP\to^{q}WP$’ $\times$ $WP \to WP$

   ‘$D,WP\to^{\mathcal{R}}WP$’ $\times D \to$ ‘$WP\to^{\mathcal{R}}WP$’

   and for each ‘$D,W\to^{\mathcal{R}}W$’ sort the corresponding ‘$W\to^{\mathcal{R}}W$’ sort exists.

   Therefore they meet the condition 3.4.1,3.

4. The sort declarations for the composition function are:

$\circ$: ‘$W\to^{p}W$’ $\times$ ‘$W\to^{q}W$’ $\to$ ‘$W\to^{s}W$’ and $\circ$: ‘$W\to^{\mathcal{R}}W$’ $\times$ ‘$W\to P$’ $\to$ ‘$W\to^{\mathcal{R}}W$’

‘$P\to^{p}W$’ $\times$ ‘$P\to^{q}W$’ $\to$ ‘$P\to^{s}W$’ ‘$W\to P$’ $\times$ ‘$W\to^{\mathcal{R}}W$’ $\to$ ‘$W\to^{\mathcal{R}}W$’

‘$D,W\to^{p}W$’ $\times$ ‘$D,W\to^{q}W$’ $\to$ ‘$D,W\to^{s}W$’ ‘$W\to P$’ $\times$ ‘$W\to P$’ $\to$ ‘$W\to P$’

where s is derived from p and q with the following matrix:

p \ q	ø	r	t	rt
ø	t	t	t	t
r	t	rt	t	rt
t	t	t	t	t
rt	t	rt	t	rt

We have to check a) whether all possible combinations are covered and

b) whether the sort declarations are associative.

a) Since there are the three different types ‘$W\to P$’, ‘$P\to^{q}W$’ and ‘$W\to^{p}W$’, there should be 3^3 combinations.

The ‘$W\to^{p}W$’ $\times$ ‘$W\to^{q}W$’ $\to$ ‘$W\to^{s}W$’ declaration stands for

‘$W\to^{p}W$’ $\times$ ‘$W\to^{q}W$’ $\to$ ‘$W\to^{s}W$’,

‘$P\to^{p}W$’ $\times$ ‘$W\to^{q}W$’ $\to$ ‘$W\to^{s}W$’ and

‘$W\to^{p}W$’ $\times$ ‘$P\to^{q}W$’ $\to$ ‘$W\to^{s}W$’.

The ‘$W\to^{\mathcal{R}}W$’ $\times$ ‘$W\to P$’ $\to$ ‘$W\to^{\mathcal{R}}W$’ declaration stands for

‘$W\to^{\mathcal{R}}W$’ $\times$ ‘$W\to P$’ $\to$ ‘$W\to^{\mathcal{R}}W$’ and

‘$P\to^{\mathcal{R}}W$’ $\times$ ‘$W\to P$’ $\to$ ‘$W\to^{\mathcal{R}}W$’.

The ‘$W\to P$’ $\times$ ‘$W\to^{q}W$’ $\to$ ‘$W\to^{q}W$’ declaration stands for

‘$W\to P$’ $\times$ ‘$W\to^{\mathcal{R}}W$’ $\to$ ‘$W\to^{\mathcal{R}}W$’ and

‘$W\to P$’ $\times$ ‘$P\to^{\mathcal{R}}W$’ $\to$ ‘$W\to^{\mathcal{R}}W$’

Together with the remaining two declarations there are in fact nine declarations.

There is only one basic type of ‘$D,W\to^{p}W$’-sorts. Therefore the single declaration scheme for these symbols is sufficient.

b) For showing the associativity we first show that the matrix is associative, i.e.

$$\forall p, q, s \in \{ø, r, t, rt\}\ (p \circ q) \circ s = p \circ (q \circ s)$$

where $\circ$ denotes the mapping represented by the matrix.

The triples p,q,s containing at least one ‘t’ or one ‘ø’ are all mapped to ‘t’

and the triples containing only ‘r’ or ‘rt’ are all mapped to ‘rt’.

Hence, the matrix is associative. (★)

Now we can show

$$\forall S_1,S_2,S_3 \in \{‘W\to^{\mathcal{R}}W’, ‘P\to^{\mathcal{R}}W’, ‘W\to P’\}\ (S_1 \times S_2) \times S_3 = S_1 \times (S_2 \times S_3)$$

All triples S_1,S_2,S_3 containing no ‘$W\to P$’ and at least one ‘$W\to^{p}W$’ are mapped to

‘$W\to^{(p\circ q)\circ s}W$’ = ‘$W\to^{p\circ(q\circ s)}W$’ (see ★)

(‘$P\to^{p}W$’ $\times$ ‘$P\to^{q}W$’) $\times$ ‘$P\to^{s}W$’ = ‘$P\to^{p}W$’ $\times$ (‘$P\to^{q}W$’ $\times$ ‘$P\to^{s}W$’) (see ★)

All triples S_1,S_2,S_3 containing at most two ‘$W\to P$’ are mapped to ‘$W\to^{\mathcal{R}}W$’ or ‘$W\to^{\mathcal{R}}P$’ respectively.

Triples S_1,S_2,S_3 consisting of three ‘$W\to P$’ are mapped to ‘$W\to P$’.

5. Since 'W$\rightarrow^{rt}$W' and 'D,W$\rightarrow^{rt}$W' are the top sorts in the corresponding parts of the sort lattice, the axioms A1-A4 characterizing $\circ$ and $\downarrow$ are sufficient to cover all instances of the axiom schemes required in def. 3.4.1,5.
6. Condition 4.1.1,2i) is fulfilled because 'W$\rightarrow^{rt}$W' and 'D,W$\rightarrow^{rt}$W' are the unique top sorts in the corresponding parts of the sort lattice.
7. The interpretation context sorts and the symbol variation function are defined (condition 4.1.1,3).

b) The translated MM-formulae are well formed CL domain $\Psi_\Sigma(\Sigma)$-formulae.
This is proved by induction on the size of MM-formulae. Let $\mathcal{G}$ be an MM-formula.
Base Case: $\mathcal{G}$ is an atom. Since MM-atoms and CL-atoms are OSPL-atoms and since the translation does not change anything $\Psi_{\mathcal{T}}(\mathcal{G})$ is a well formed CL domain atom (def. 4.1.3,i).
Induction step: Let $\mathcal{G}$ be a non atomic MM-formula.
The induction hypothesis states that all formulae of smaller size than $\mathcal{G}$ are translated into well formed CL domain formulae. We perform a case analysis according to the structure of $\mathcal{G}$.
If $\mathcal{G}$'s top operator is a classical logical connective or quantifier then the induction hypothesis is immediately applicable. The interesting cases are the modal operators.
Cases $\mathcal{G} = \Box^{\mathcal{R}}\mathcal{F}$ and $\mathcal{G} = \Diamond^{\mathcal{R}}\mathcal{F}$
The translation rules are $\Psi_{\mathcal{T}}(\mathcal{G}) = \forall(\exists)$x:'W$\rightarrow^{\mathcal{R}}$W' $\Psi_{\mathcal{T}}(\mathcal{F})$.
$\Psi_{\mathcal{T}}(\mathcal{F})$ is well formed according to the induction hypothesis and since 'W$\rightarrow^{\mathcal{R}}$W' is a context sort in $\Psi_\Sigma(\Sigma)$, $\Psi_{\mathcal{T}}(\mathcal{G})$ is well formed (def. 4.1.3,v).
Cases $\mathcal{G} = [\![t]\!]^{\mathcal{R}}\mathcal{F}$ and $\mathcal{G} = \langle t\rangle^{\mathcal{R}}\mathcal{F}$
The translation rules are $\Psi_{\mathcal{T}}(\mathcal{G}) = \forall(\exists)\downarrow$(x:'D,W$\rightarrow^{\mathcal{R}}$W', z:S(t)=t) $\Psi_{\mathcal{T}}(\mathcal{F})$.
$\Psi_{\mathcal{T}}(\mathcal{F})$ is well formed according to the induction hypothesis. Since 'D,W$\rightarrow^{\mathcal{R}}$W' is a context sort and since x and z are new variables not occurring in $\Psi_{\mathcal{T}}(\mathcal{F})$, $\Psi_{\mathcal{T}}(\mathcal{G})$ is well formed (def. 4.1.3,iv).
Cases $\mathcal{G} = \blacktriangleright\mathcal{F}$ and $\mathcal{G} = \twoheadrightarrow\mathcal{F}$
The translation rules are $\Psi_{\mathcal{T}}(\mathcal{G}) = \forall(\exists)$ p:'W$\rightarrow$P' $\exists(\forall)$ x:'P$\rightarrow^{rt}$W' $\Psi_{\mathcal{T}}(\mathcal{F})$.
$\Psi_{\mathcal{T}}(\mathcal{F})$ is well formed according to the induction hypothesis.'W$\rightarrow$P' as well as 'P$\rightarrow^{rt}$W' are both context sorts. According to def. 4.1.3,v, $\Psi_{\mathcal{T}}(\mathcal{G})$ is well formed.
Case $\mathcal{G} = (|t)\mathcal{F}$
The translation rule is $\Psi_{\mathcal{T}}((|t)\mathcal{F}) = \forall$p:'W$\rightarrow$P' $\wp$(BF(t) $\circ$ +1) $\Psi_{\mathcal{T}}(\mathcal{F})$.
$\Psi_{\mathcal{T}}(\mathcal{F})$ is well formed according to the induction hypothesis.
Since t is a domain term and the sort of BF is D $\rightarrow$ 'P$\rightarrow^{rt}$W', the sort of BF(t) is 'P$\rightarrow^{rt}$W'. The sort of +1 is 'P$\rightarrow^{\wp}$W'. Thus, the sort of (BF(t) $\circ$ +1) is 'P$\rightarrow^{rt}$W' which is a functional context sort. According to def. 4.1.3,vii, $\wp$(BF(t) $\circ$ +1) $\Psi_{\mathcal{T}}(\mathcal{F})$ is a well formed CL context formula and according to def. 4.1.3,v, $\Psi_{\mathcal{T}}(\mathcal{G})$ is well formed.
Cases $\mathcal{G} = \mathcal{F}\forall\mathrm{U}\,\mathcal{G}$, $\mathcal{G} = \mathcal{F}\forall\mathrm{U}^{r}\,\mathcal{G}$, $\mathcal{G} = \mathcal{F}\exists\mathrm{U}\,\mathcal{G}$ and $\mathcal{G} = \mathcal{F}\exists\mathrm{U}^{r}\,\mathcal{G}$
The translation rules are
$\Psi_{\mathcal{T}}(\mathcal{G}) = \forall(\exists)$ p:'W$\rightarrow$P' $\exists$x:'P$\rightarrow^{rt}$W' ($\Psi_{\mathcal{T}}(\mathcal{G}) \wedge \forall$y:'P$\rightarrow^{rt}$W'-x ($<(\leq)$(p$\circ$y, p$\circ$x) $\Rightarrow \Psi_{\mathcal{T}}(\mathcal{F})$)).
$\Psi_{\mathcal{T}}(\mathcal{G})$ and $\Psi_{\mathcal{T}}(\mathcal{F})$ are well formed according to the induction hypothesis. $<$ and $\leq$ are context predicates of sort WP$\times$WP. p$\circ$y and p$\circ$x are both terms of sort 'W$\rightarrow^{rt}$W' therefore, according to the replacement rule in def. 4.1.3,ii, $<$(p$\circ$y, p$\circ$x) and $\leq$(p$\circ$y, p$\circ$x) are well formed CL domain atoms. According to def. 4.1.3,vi, $\forall$y:'P$\rightarrow^{rt}$W'-x ($<(\leq)$(p$\circ$y, p$\circ$x) $\Rightarrow \Psi_{\mathcal{T}}(\mathcal{F})$) is a well formed domain formula. And finally, since 'W$\rightarrow$P' and 'P$\rightarrow^{rt}$W' are functional context sorts $\Psi_{\mathcal{T}}(\mathcal{G})$ is well formed according to def. 4.1.3,v.

c) The axioms generated by $\Psi_{\mathcal{S}}(\mathcal{S})$ are well formed CL context $\Psi_\Sigma(\Sigma)$-formulae.This is a straightforward check.
(A syntax error in the translation rules may lead to an erraneous implementation of the translation algorithm. Here we have to be careful. A syntax error in the axioms is usually detected by a properly implemented formula parser. Therefore we can afford to be lax about this.) ■

Derived Formulae

The specification morphism generates a number of CL formulae to axiomatize labeled possible worlds structures. To confirm that the axioms really describe our intuition about the possible worlds structures and to provide a better basis for an implementation we derive some useful lemmas.

A5 $\forall x,y,z{:}\text{‘}W \rightarrow^{rt} W\text{’}\ (x \circ y) \circ z = x \circ (y \circ z)$ (associativity of $\circ$)
A6 $\forall x,y,z{:}\text{‘}D,W \rightarrow^{rt} W\text{’}\ (x \circ y) \circ z = x \circ (y \circ z)$
Proof: Lemma 3.4.2

B3 $\forall x{:}\text{‘}W \rightarrow^{rt} W\text{’}\ \ x \circ ID = x$
Proof: $\downarrow(x \circ ID, w) = \downarrow(ID, \downarrow(x, w))$ (A3)
$= \downarrow(x, w)$ (B1)
$\Rightarrow\ x \circ ID = x$ (A1)

B4 $\forall x{:}\text{‘}W \rightarrow^{rt} W\text{’}\ \ ID \circ x = x$
Proof: $\downarrow(ID \circ x, w) = \downarrow(x, \downarrow(ID, w))$ (A3)
$= \downarrow(x, w)$ (B1)
$\Rightarrow\ x \circ ID = x$ (A1)

B5 $\forall x{:}\text{‘}W \rightarrow^{rt} W\text{’}\ \forall l{:}D\quad x \circ \downarrow(IDL, l) = x$
Proof: $\downarrow(x \circ \downarrow(IDL, l), w) = \downarrow(\downarrow(IDL, l), \downarrow(x, w))$ (A3)
$= \downarrow(x, w)$ (B2)
$\Rightarrow\ x \circ \downarrow(IDL, l) = x$ (A1)

B6 $\forall x{:}\text{‘}W \rightarrow^{rt} W\text{’}\ \forall l{:}D\quad \downarrow(IDL, l) \circ x = x$
Proof: $\downarrow(\downarrow(IDL, l) \circ x, w) = \downarrow(x, \downarrow(\downarrow(IDL, l), w))$ (A3)
$= \downarrow(x, w)$ (B2)
$\Rightarrow\ \downarrow(IDL, l) \circ x = x$ (A1)

C4 $\forall p{:}\text{‘}W \rightarrow P\text{’}\ w{:}WP\ w = (p \circ PA(w))(w))$
Proof: $\forall p{:}\text{‘}W \rightarrow P\text{’}\ w{:}WP\ PW(w) = PW(p(w)) \Rightarrow\ w = PA(w)(p(w))$ (C3)
$\Rightarrow\ \forall p{:}\text{‘}W \rightarrow P\text{’}\ w{:}WP\ PW(w) = PW(w) \Rightarrow\ w = PA(w)(p(w))$ (K1 below)
$\Rightarrow\ \forall p{:}\text{‘}W \rightarrow P\text{’}\ w{:}WP\ \ w = PA(w)(p(w))$ (reflexivity of =)
$\Rightarrow\ \forall p{:}\text{‘}W \rightarrow P\text{’}\ w{:}WP\ \ w = (p \circ PA(w))(w))$ (A3)

D6 $\forall x{:}\text{‘}P \rightarrow^{rt} W\text{’}\ \forall p{:}\text{‘}W \rightarrow P\text{’}\ \forall w{:}WP\ PW(MP(x \circ z \circ p) \circ x)(w)) = PW(x(w))$
Proof: $x(w) = x(w) \Rightarrow PW(MP(x \circ z \circ p) \circ x)(w)) = PW(x(w))$ (D3)
$\Rightarrow\ PW(MP(x \circ z \circ p) \circ x)(w)) = PW(x(w))$ (Reflexivity of =)

D7 $\forall x,y{:}\text{‘}P \rightarrow^{rt} W\text{’}\ \forall w{:}WP\ \ PW(x(w)) = PW(y(w)) \Rightarrow x(w) = y(w)$
Proof: $PW(MP(x \circ ID \circ ID) \circ y)(w)) = PW(x(w)) \Rightarrow x(w) = y(w)$ (D3)
$\Rightarrow\ PW(y(w)) = PW(x(w)) \Rightarrow x(w) = y(w)$ (B3, B4, D2)

D8 $\forall x,y{:}\text{‘}P \rightarrow^{rt} W\text{’}\ \forall p{:}\text{‘}W \rightarrow P\text{’}\ \forall w{:}WP$
$x(w) = y(w) \Rightarrow PW(MP(x \circ z \circ p) \circ x)(w)) = PW(MP(x \circ z \circ p) \circ y)(w))$
Proof: Suppose $x(w) = y(w)$
$\Rightarrow\ PW(MP(x \circ z \circ p) \circ y)(w)) = PW(x(w))$ (D4)
$\Rightarrow\ PW(MP(x \circ z \circ p) \circ x)(w)) = PW(MP(x \circ z \circ p) \circ y)(w))$ (D6)

D9 $\forall x,y,z$:‘$P \to^{rt} W$’ $\forall p$:‘$W \to P$’ $\forall w{:}WP$

$x(w) < y(w) \Rightarrow (MP(x \circ z \circ p) \circ x)(w) < (MP(x \circ z \circ p) \circ y)(w)$

Proof: Let $x(w) < y(w)$ $(\ast)$

Assume $(MP(x \circ z \circ p) \circ y)(w) \le (MP(x \circ z \circ p) \circ x)(w)$

$\Rightarrow$ $y(MP(x \circ z \circ p)(w) \le x(MP(x \circ z \circ p)(w)$ (A3)

$\Rightarrow$ $(y \circ \rightarrowtail(y, x))(MP(x \circ z \circ p)(w) = x(MP(x \circ z \circ p)(w)$ (I2)

$\Rightarrow$ $MP(x \circ z \circ p) \circ y \circ \rightarrowtail(y, x))(w) = (MP(x \circ z \circ p) \circ x)(w)$ (A3)

$\Rightarrow$ $PW(MP(x \circ z \circ p) \circ y \circ \rightarrowtail(y, x))(w)) = PW(x(w))$ (D6)

$\Rightarrow$ $(y \circ \rightarrowtail(y, x))(w) = x(w)$ (D4)

$\Rightarrow$ $y(w) \le x(w)$ (I1)

$\Rightarrow$ $\neg\, x(w) < y(w)$ (H5)

$\Rightarrow$ contradiction $(\ast)$

$\Rightarrow$ $(MP(x \circ z \circ p) \circ x)(w) < (MP(x \circ z \circ p) \circ y)(w)$ (H5)

D10 $\forall x,y,z$:‘$P \to^{rt} W$’ $\forall p$:‘$W \to P$’ $\forall w{:}WP$

$(MP(x \circ z \circ p) \circ x)(w) < (MP(x \circ z \circ p) \circ y)(w) \Rightarrow x(w) < y(w)$

Proof: Let $(MP(x \circ z \circ p) \circ x)(w) < (MP(x \circ z \circ p) \circ y)(w)$

Assume $y(w) \le x(w)$

$\Rightarrow$ $(y \circ \rightarrowtail(y, x))(w) = x(w)$ (I2)

$\Rightarrow$ $PW(MP(x \circ z \circ p) \circ y \circ \rightarrowtail(y, x))(w)) = PW((MP(x \circ z \circ p) \circ x)(w))$ (D7)

$\Rightarrow$ $MP(x \circ z \circ p) \circ y)(w) \le (MP(x \circ z \circ p) \circ x)(w)$ (I1)

$\Rightarrow$ $\neg\, MP(x \circ z \circ p) \circ x)(w) < (MP(x \circ z \circ p) \circ y)(w)$ (H5)

$\Rightarrow$ contradiction

$\Rightarrow$ $\neg\, y(w) \le x(w)$

$\Rightarrow$ $x(w) < y(w)$ (H1)

D11 $\forall x,y,z$:‘$P \to^{rt} W$’ $\forall p$:‘$W \to P$’ $\forall w{:}WP$

$(MP(x \circ z \circ p) \circ y)(w) < (MP(x \circ z \circ p) \circ x)(w) \Rightarrow y(w) < x(w)$

Proof: Suppose $(MP(x \circ z \circ p) \circ y)(w) < (MP(x \circ z \circ p) \circ x)(w)$

$\Rightarrow$ $y(MP(x \circ z \circ p)(w) < x(MP(x \circ z \circ p)(w)$ (A3)

$\Rightarrow$ $(y \circ \text{-}R(\rightarrowtail(y, x)))(MP(x \circ z \circ p)(w) = x(MP(x \circ z \circ p)(w)$ (I3 below)

$\Rightarrow$ $MP(x \circ z \circ p) \circ y \circ \text{-}R(\rightarrowtail(y, x)))(w) = (MP(x \circ z \circ p) \circ x)(w)$ (A3)

$\Rightarrow$ $PW(MP(x \circ z \circ p) \circ y \circ \text{-}R(\rightarrowtail(y, x)))(w)) = PW(x(w))$ (D6)

$\Rightarrow$ $(y \circ \text{-}R(\rightarrowtail(y, x)))(w) = x(w)$ (D4)

$\Rightarrow$ $y(w) < x(w)$ (I4)

D12 $\forall x,y,z$:‘$P \to^{rt} W$’ $\forall p$:‘$W \to P$’ $\forall w{:}WP$ $x(w) = y(w) \Rightarrow (MP(x \circ z \circ p) \circ x)(w) = (MP(x \circ z \circ p) \circ y)(w)$

Proof: Let $x(w) = y(w)$ $(\ast)$

Assume $(MP(x \circ z \circ p) \circ x)(w) \neq (MP(x \circ z \circ p) \circ y)(w)$

$\Rightarrow (MP(x \circ z \circ p) \circ x)(w) < (MP(x \circ z \circ p) \circ y)(w) \vee$

$(MP(x \circ z \circ p) \circ y)(w) < (MP(x \circ z \circ p) \circ x)(w)$ (H6)

Suppose $(MP(x \circ z \circ p) \circ x)(w) < (MP(x \circ z \circ p) \circ y)(w)$

$\Rightarrow$ $x(w) < y(w)$ (D10)

$\Rightarrow$ contradiction (H2, $\ast$)

$\Rightarrow$ $(MP(x \circ z \circ p) \circ y)(w) < (MP(x \circ z \circ p) \circ x)(w)$

$\Rightarrow$ $y(w) < x(w)$ (D11)

$\Rightarrow$ contradiction (H2, $\ast$)

$\Rightarrow$ $(MP(x \circ z \circ p) \circ x)(w) = (MP(x \circ z \circ p) \circ y)(w)$

D13 $\forall x,y,z$:‘$P\to^{rt}W$’ $\forall p$:‘$W\to P$’ $\forall w$:WP $x(w) \le y(w) \Rightarrow (MP(x \circ z \circ p) \circ x)(w) \le (MP(x \circ z \circ p) \circ y)(w)$

Proof: Let $x(w) \le y(w)$

Assume $(MP(x \circ z \circ p) \circ y)(w) < (MP(x \circ z \circ p) \circ x)(w)$

$\Rightarrow$ $y(w) < x(w)$ (D11)

$\Rightarrow$ $\neg\, x(w) \le y(w)$ (H5)

$\Rightarrow$ contradiction

$\Rightarrow$ $\neg\, (MP(x \circ z \circ p) \circ y)(w) < (MP(x \circ z \circ p) \circ x)(w)$

$\Rightarrow$ $(MP(x \circ z \circ p) \circ x)(w) \le (MP(x \circ z \circ p) \circ y)(w)$ (H1)

F6 $\forall x$:‘$P\to^{rt}W$’ $\forall w$:WP $x(w) = w \lor x(w) = +1(w)$

Proof: $x(w) = w \lor x(w) = -R(x)(w)$ (F2)

$\Rightarrow x(w) = w \lor x(w) = +1(w)$ (F1, -R:‘$P\to^{rt}W$’ $\to$ ‘$P\to^{\emptyset}W$’)

G5 $\forall x$:‘$P\to^{rt}W$’ $\forall w$:WP $w \le x(w)$

Proof: $w_2 = x(w) \Rightarrow w \le w_2$ (I1)

$\Rightarrow$ $w \le x(w)$ (reflexivity of =)

H4 $\forall w_1,w_2$:WP $w_1 < w_2 \Rightarrow \neg\, w_2 < w_1$

Proof: Let $w_1 < w_2$

$\Rightarrow$ $w_1 \ne w_2$ (H2)

and $w_1 \le w_2$ (✛) (H1)

Assume $w_2 < w_1$

$\Rightarrow$ $w_2 \le w_1$ (H1)

$\Rightarrow$ $w_1 = w_2$ (H3, ✛)

$\Rightarrow$ contradiction

$\Rightarrow$ $\neg\, w_2 < w_1$

H5 $\forall w_1,w_2$:WP $w_1 < w_2 \Rightarrow \neg\, w_2 \le w_1$

Proof: Let $w_1 < w_2$

$\Rightarrow$ $w_1 \ne w_2$ (H2)

and $w_1 \le w_2$ (✛) (H1)

Assume $w_2 \le w_1$

$\Rightarrow$ $w_1 = w_2$ (H3, ✛)

$\Rightarrow$ contradiction

$\Rightarrow$ $\neg\, w_2 \le w_1$

H6 $\forall x,y$:‘$P\to^{rt}W$’ $\forall w$:WP $x(w) \ne y(w) \Rightarrow x(w) < y(w) \lor y(w) < x(w)$

Proof: Suppose $x(w) \ne y(w)$ (✢)

$\Rightarrow$ $\neg x(w) \le y(w) \lor x(w) < y(w)$ (H3)

$\Rightarrow$ $y(w) \le x(w) \lor x(w) < y(w)$ (G4)

$\Rightarrow$ $y(w) < x(w) \lor y(w) = x(w) \lor x(w) < y(w)$ (H3)

$\Rightarrow$ $x(w) < y(w) \lor y(w) < x(w)$ (✢)

I3 $\forall x,y$:‘$P\to^{rt}W$’ $\forall w$:WP $<(x(w), y(w)) \Rightarrow (x \circ -R(➛(x, y)))(w) = y(w)$.

Proof: Suppose $<(x(w), y(w))$

$\Rightarrow$ $x(w) \ne y(w)$ (✢) (H2)

and $\le(x(w), y(w))$ (H1)

$\Rightarrow$ $(x \circ ➛(x, y))(w) = y(w)$ (✸) (I2)

$\Rightarrow$ $➛(x, y)(x(w)) = y(w)$ (✶) (A3)

$\Rightarrow$ $\rightarrowtail(x, y)(x(w)) = x(w) \vee \rightarrowtail(x, y)(x(w)) = -R(\rightarrowtail(x, y))(x(w)))$ (F2)
$\Rightarrow$ $y(w) = x(w) \vee \rightarrowtail(x, y)(x(w)) = -R(\rightarrowtail(x, y))(x(w)))$ ($\ast$)
$\Rightarrow$ $\rightarrowtail(x, y)(x(w)) = -R(\rightarrowtail(x, y))(x(w)))$ ($\div$)
$\Rightarrow$ $(x \circ \rightarrowtail(x, y))(w) = (x \circ -R(\rightarrowtail(x, y)))(w))$ (A3)
$\Rightarrow$ $(x \circ -R(\rightarrowtail(x, y)))(w) = y(w)$ ($\ast$)

I4 $\forall x,y$:‘P→$^{\text{nt}}$W’ $\forall w$:WP $\forall z$:‘P→$^{\text{t}}$W’ w:WP $(x \circ z)(w) = y(w) \Rightarrow x(w) < y(w)$
Proof: Suppose $(x \circ z)(w) = y(w)$
$\Rightarrow$ $x(w) \leq y(w)$ ($\ast$) (I1)
and $z(x(w)) \neq x(w)$ (F3)
and $z(x(w)) \neq y(w)$ (A3)
$\Rightarrow$ $x(w) \neq y(w)$
$\Rightarrow$ $x(w) < y(w)$ (H3, $\ast$)

K1 $\forall p$:‘W→P’ $\forall w$:WP PW(p(w)) = PW(w)
Proof: $ID(w) \leq ID(w) \Rightarrow PW(MP(ID \circ p) \circ ID)(w)) = PW(ID(w))$ (D4, x = y = ID)
$\Rightarrow$ $PW(MP(ID \circ p) \circ ID)(w)) = PW(ID(w))$ (G1)
$\Rightarrow$ $PW(MP(p)(w)) = PW(w)$ (B4, B3, B1)
$\Rightarrow$ $PW(p(w)) = PW(w)$ (C1)

The interpretation morphism $\Psi_{\mathfrak{I}}$, whose existence confirms soundness and completeness of the translation, has to translate the relational description of the accessibility relation into a functional description where the argument-value relation of the context access functions represents the transitions in the possible worlds structure.

Definition 5.2.4 The Interpretation Morphism $\Psi_{\mathfrak{I}}$

Given an MM-interpretation $\mathfrak{I} = ((\mathcal{W}, \mathfrak{R}), \mathcal{V}, \mathcal{W}_0)$ over the MM-signature Σ, the interpretation morphism $\Psi_{\mathfrak{I}}$ generates the following CL-interpretation $\mathfrak{I}_{CL} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, (<\mathcal{W}_0, \mathcal{P}_0>), \mathcal{P})$ (def. 4.2.2) over $\Psi_\Sigma(\Sigma)$ where

1. $\mathcal{C}$ is a functional $\Psi_\Sigma(\Sigma)_C$-structure where the context symbols are interpreted as follows:
   Interpretation of the non functional sort symbols:
   $D_{\mathcal{C}} := D_{\mathcal{W}_0}$ for all domain sorts D. (the domains are equal in all worlds)
   $W_{\mathcal{C}} := \mathcal{W}$
   $WP_{\mathcal{C}} := \{<\mathcal{W}, \mathcal{P}> \mid \mathcal{W} \in W_{\mathcal{C}}, \mathcal{P} \in \mathfrak{P}(\mathcal{W}_0) \text{ and } \mathcal{W} \in \mathcal{P}\}$
   $\text{‘ID’}_{\mathcal{C}} := \{\text{identity function on } WP_{\mathcal{C}} \rightarrow WP_{\mathcal{C}}\}$

   In the sequel we write the composition function in $\mathcal{C}$ also as $\circ$.

   For the definition of the context access functions we need some auxiliary functions “$+n$” which move on each path exactly n steps forward:
   For $n \geq 0$ let $+n$: $WP_{\mathcal{C}} \rightarrow WP_{\mathcal{C}}$ with $+n(<\mathcal{W}_m, \mathcal{P}>) = <\mathcal{W}_{m+n}, \mathcal{P}>$,
   $\mathcal{W}_m$ and $\mathcal{W}_{m+n}$ being the m´th and m+n´th worlds in $\mathcal{P}$ (see lemma 5.1.7).
   Let $(+n \circ +m)(w) := +(n+m)(w)$. Since $\mathfrak{R}^{\emptyset}$ is serial, all these $+n$ functions are total.

   As a notational convention we write projection of WP-tuples w to the world component with $w_{|W}$ and projection to the path component with $w_{|P}$.
   Furthermore we use $w_{\approx P} := \{w'_{|W} \mid w'_{|P} = w_{|P}\}$ to denote the set of worlds on a path.

Interpretation of the functional sort symbols:

$\text{'W}\rightarrow\text{P'}_C$ $:= \{p{:}\mathrm{WP}_C\rightarrow\mathrm{WP}_C \mid p(<\mathcal{W}, \mathcal{P}>)_{|\mathrm{W}} = \mathcal{W}, p \text{ is total}\}$

$\text{'P}\rightarrow^{\mathcal{R}}\text{W'}_C$ $:= \{\chi{:}\mathrm{WP}_C\rightarrow\mathrm{WP}_C \mid \forall w \in \mathrm{WP}_C\ \exists \mathrm{n} \in \mathcal{N}^{\mathcal{R}}\ \chi(w) = +n(w)$
and $\forall w'$ with $w'_{|\mathrm{W}} = w_{|\mathrm{W}}$ and $\chi(w)_{|\mathrm{W}} \in w'_{\approx\mathrm{P}}$: $\chi(w') = +n(w')\}$

($\mathcal{N}^{\mathcal{R}}$ is defined in lemma 5.1.7,b. In particular $\text{'P}\rightarrow^{\emptyset}\text{W'}_C = \{+1\}$)

$\text{'W}\rightarrow^{\mathcal{R}}\text{W'}_C$ $:= \{\chi{:}\mathrm{WP}_C\rightarrow\mathrm{WP}_C \mid \exists\chi_{\mathrm{P}} \in \text{'W}\rightarrow\text{P'}_C\ \exists\chi_{\mathrm{W}} \in \text{'P}\rightarrow^{\mathcal{R}}\text{W'}_C\ \chi = \chi_{\mathrm{P}} \circ \chi_{\mathrm{W}}\}$

$\text{'D,W}\rightarrow^{\mathcal{R}}\text{W'}_C := \{\chi{:}\mathrm{D}_C\times\mathrm{WP}_C\rightarrow\mathrm{WP}_C \mid \forall \ell\in\mathrm{D}_C\ \chi(\ell) \in \text{'W}\rightarrow^{\mathcal{R}}\text{W'}_C \text{ and } \forall w\in\mathrm{WP}_C\ (w_{|\mathrm{W}},\chi(\ell)(w)_{|\mathrm{W}}) \in \Re^{\mathcal{R}}(\ell)\}$

Interpretation of the constant symbols:

ID_C is the identity function on $\mathrm{WP}_C\rightarrow\mathrm{WP}_C$

IDL_C is the identity function on $\mathrm{D}_C\times\mathrm{WP}_C\rightarrow\mathrm{WP}_C$

$+1_C = +1$ is the single element of $\text{'P}\rightarrow^{\emptyset}\text{W'}_C$

$+1\mathrm{L}_C := +1L \in \text{'D,W}\rightarrow^{\emptyset}\text{W'}_C$ with

$\forall\ell\in\mathrm{D}_C$ $+1L(\ell)(w) = +1(w)$ in case $w = \mathrm{BF}_C(\ell)(w_0)$ for some $w_0 \in \mathrm{WP}_C$
$+1L(\ell)(w) = \chi(\ell)(w)$ otherwise, where χ is some element of $\text{'D,W}\rightarrow^{\emptyset}\text{W'}_C$ (uninteresting case)

Interpretation of the function symbols:

$\forall<\mathcal{W},\mathcal{P}> \in \mathrm{WP}_C$
$\mathrm{PW}_C(<\mathcal{W},\mathcal{P}>) = \mathcal{W}$, i.e. $\mathrm{PW}_C = {}_{|\mathrm{W}}$.

$\forall\ell\in\mathrm{D}_C$
$\mathrm{BF}_C(\ell) = \chi$ where $\chi\in \text{'P}\rightarrow^{\mathrm{rt}}\text{W'}_C$ and $\forall w\in\mathrm{WP}_C\ (\chi(w)_{|\mathrm{W}}, (\chi\circ+1)(w))_{|\mathrm{W}}) \in \Re^{\emptyset}(\ell)$
and $\chi(w)_{|\mathrm{W}}$ is the first world among the $y(w)_{\mathrm{W}}$ with $(y(w)_{|\mathrm{W}}, (y\circ+1)(w))_{|\mathrm{W}}) \in \Re^{\emptyset}(\ell)$.

$\forall w\in\mathrm{WP}_C$
$\mathrm{PA}_C(w) = p$ where $p\in\text{'W}\rightarrow\text{P'}_C$ and $\forall w'\in\mathrm{WP}_C$ if $w'_{|\mathrm{W}} = w_{|\mathrm{W}}$ then $p(w') = w$ else $p(w') = w'$

$\forall\chi\in\text{'W}\rightarrow^{\mathrm{rt}}\text{W'}_C$
$\mathrm{MP}_C(\chi) \in \text{'W}\rightarrow\text{P'}_C$ with $\mathrm{MP}_C(\chi)(<\mathcal{W},\mathcal{P}>) = <\mathcal{W}, \chi(<\mathcal{W},\mathcal{P}>)_{|\mathrm{P}}>$.

$\forall\chi,y\in\text{'W}\rightarrow^{\mathrm{rt}}\text{W'}_C$
$\mathrm{MW}_C(\chi) \in \text{'P}\rightarrow^{\mathcal{R}}\text{W'}_C$
with $\mathrm{MW}_C(\chi)(<\mathcal{W}, (\chi\circ y)(<\mathcal{W},\mathcal{P}>)_{|\mathrm{P}}>) = <\chi(<\mathcal{W},\mathcal{P}>)_{|\mathrm{W}}, (\chi\circ y)(<\mathcal{W},\mathcal{P}>)_{|\mathrm{P}}>)$.
otherwise $\mathrm{MW}_C(\chi)(<\mathcal{W},\mathcal{P}>) = <\chi(<\mathcal{W},\mathcal{P}>_{|\mathrm{W}}, \mathcal{P}>)$

$\forall\chi\in\text{'W}\rightarrow^{\mathrm{rt}}\text{W'}_C\ \forall y\in\text{'D,W}\rightarrow^{\mathrm{rt}}\text{W'}_C$
$\mathrm{LT}_C(\chi, y) \in \text{'D,W}\rightarrow^{\mathrm{rt}}\text{W'}_C$ such that $\forall\ell\in\mathrm{D}_C$: $\chi\circ y(\ell) = \mathrm{LT}_C(\chi, y)(\ell)$

$\forall\chi\in\text{'X}\rightarrow^{\mathrm{r}y}\text{W'}_C$ where $\mathrm{X} \in \{\mathrm{W}, \mathrm{P}\}$ and $y \in \{\text{""}, \mathrm{t}\}$
$\cdot\mathrm{R}_C(\chi) = y$ where $y\in\text{'X}\rightarrow^{y}\text{W'}_C$ and $\forall w\in\mathrm{WP}_C$ such that
if $\chi(w) \neq w$ then $\chi(w) = y(w)$ else $\chi(w) = +1(w)$.

$\forall\chi\in\text{'P}\rightarrow^{\mathrm{t}}\text{W'}_C$ $\forall w\in\mathrm{WP}_C$
if $\chi(w) = +(n+1)(w)$ for some $\mathrm{n}>0$ then $\mathrm{FS}_C(\chi)(w) = +n(w)$
else $\mathrm{FS}_C(\chi)(w) = +1(w)$ (irrelevant case)

$\forall\chi\in\text{'P}\rightarrow^{\mathrm{rt}}\text{W'}_C \setminus \text{'P}\rightarrow^{\mathrm{t}}\text{W'}_C\ \forall w\in\mathrm{WP}_C$
if $\chi(w) = +(n+1)(w)$ for some $\mathrm{n}\geq 0$ then $\mathrm{FS}_C(\chi)(w) = +n(w)$
else $\mathrm{FS}_C(\chi)(w) = w$ (irrelevant case)

$\forall\chi,y\in\text{'P}\rightarrow^{\mathrm{rt}}\text{W'}_C\ \forall w\in\mathrm{WP}_C$
if $\chi(w) = +n(w)$ for some n and
$y(w) = +m(w)$ for some $\mathrm{m}\geq\mathrm{n}$ then ➛$_C(\chi, y)(w) = +(m-n)(w)$
else ➛$_C(\chi, y)(w) = w$ (irrelevant case)

Interpretation of the predicate symbols

$\leq_C := \{(w_1, w_2) \mid w_1, w_2 \in \mathrm{WP}_C \text{ and } w_2 = +n(w_1) \text{ for some } \mathrm{n}\geq 0\}$

$<_C := \{(w_1, w_2) \mid w_1, w_2 \in \mathrm{WP}_C \text{ and } w_2 = +n(w_1) \text{ for some } \mathrm{n} > 0\}$

2. $\mathcal{SV}$ maps the tuple $<\mathcal{W}, \mathcal{P}>$ to its element $\mathcal{W}$ which is a $\Pi_\Sigma(\Sigma)_D$-structure.
3. The path $\mathcal{P}_0$ in the initial context of $\Im_{CL}$ as well as the $\mathcal{P}$-component are irrelevant for the interpretation of closed formulae and may be chosen at random.

The **inverse interpretation morphism** Ψ_3^{-1} generates from the CL-interpretation $\Im_{CL} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, (w_0), \mathcal{P})$ the MM-interpretation $\Im = ((W_\mathcal{C}, \Re), \mathcal{V}, PW_\mathcal{C}(w_0))$ where

a) $\Re^\mathcal{R} := \{(PW_\mathcal{C}(w), PW_\mathcal{C}(\chi(w))) \mid \chi \in \text{'P}\to^\mathcal{R}\text{W'}_\mathcal{C}\ w \in WP_\mathcal{C}\}$
For $w \in WP_\mathcal{C}$ let $\mathcal{P}(w) := \{PW_\mathcal{C}(\chi(w)) \mid \chi \in \text{'P}\to^{\pi}\text{W'}_\mathcal{C}\}$. $\mathcal{P}(w)$ is the path described by w.

b) $\mathbf{P}(\mathcal{W}) := \{\mathcal{P}(w) \mid PW_\mathcal{C}(w) = \mathcal{W}\}$

c) A transition $\Re^\mathcal{R}(\mathcal{W}_1, \mathcal{W}_2)$ is labeled with a label ℓ
iff $\exists\ \chi \in \text{'D,W}\to^\mathcal{R}\text{W'}_\mathcal{C}\ \exists w \in WP_\mathcal{C}$ with $\mathcal{W}_1 = PW_\mathcal{C}(w)$ and $\mathcal{W}_2 = PW_\mathcal{C}(\chi(\ell, w))$ ■

The next lemmas confirm that the interpretations of the functional context sorts are rich enough to reach all accessible worlds and paths.

Lemma 5.2.5 The 'W→P'-Functions Reach all Path Crossing a World.
Given an MM-interpretation $\Im = ((\mathcal{W}, \Re), \mathcal{V}, \mathcal{W}_0)$ and the translated interpretation $\Im_{CL}$,
$\forall \mathcal{W} \in \mathcal{W}, \mathcal{P}_0 \in \mathbf{P}(\mathcal{W}_0)$ with $\mathcal{W} \in \mathcal{P}_0$: $\mathcal{P} \in \mathbf{P}(\mathcal{W}) \Leftrightarrow \exists\ p \in \text{'W}\to\text{P'}_\mathcal{C}\ p(<\mathcal{W},\mathcal{P}_0>) = <\mathcal{W},\mathcal{P}>$.
Proof: "⇒"
The function p with $p(<\mathcal{W},\mathcal{P}_0>) = <\mathcal{W},\mathcal{P}>$ and $p(w) = w$ for all $w \neq <\mathcal{W},\mathcal{P}_0>$ is total and therefore in 'W→P'$_\mathcal{C}$
"⇐" Since $p \in WP_\mathcal{C} \to WP_\mathcal{C}$, $\mathcal{W} \in \mathcal{P}$ and therefore $\mathcal{P} \in \mathbf{P}(\mathcal{W})$. ■

Lemma 5.2.6 The 'P→$^\mathcal{R}$W'-Functions Reach all $\Re^\mathcal{R}$-Accessible Worlds on a Path.
Given an MM-interpretation $\Im = ((\mathcal{W}, \Re), \mathcal{V}, \mathcal{W}_0)$ and the translated interpretation $\Im_{CL}$,
$\forall \mathcal{W} \in \mathcal{W}, \mathcal{P} \in \mathbf{P}(\mathcal{W})$: $(\mathcal{W}_1 \in \mathcal{P}$ and $\mathcal{W}_2 \in \mathcal{P}$ and $\Re^\mathcal{R}(\mathcal{W}_1, \mathcal{W}_2)) \Leftrightarrow \exists\ \chi \in \text{'W}\to^\mathcal{R}\text{P'}_\mathcal{C}\ \chi(<\mathcal{W}_1,\mathcal{P}>) = <\mathcal{W}_2,\mathcal{P}>$.
Proof: "⇒" $\Re^\mathcal{R}(\mathcal{W}_1, \mathcal{W}_2)$ implies there is an $n \in \mathcal{N}^\mathcal{R}$ such that $\mathcal{W}_2$ is reached from $\mathcal{W}_1$ in n $\Re^\varnothing$-transitions on $\mathcal{P}$ (lemma 5.1.7,b). We simply select $\chi := +n \in \text{'W}\to^\mathcal{R}\text{P'}_\mathcal{C}$
"⇐" $\chi(<\mathcal{W}_1,\mathcal{P}>) = +n(<\mathcal{W}_1,\mathcal{P}>)$ for some $n \in \mathcal{N}^\mathcal{R}$.
Hence, $\mathcal{W}_1 \in \mathcal{P}$ and $\mathcal{W}_2 \in \mathcal{P}$ and $\Re^\mathcal{R}(\mathcal{W}_1, \mathcal{W}_2)$ (lemma 5.1.7,b). ■

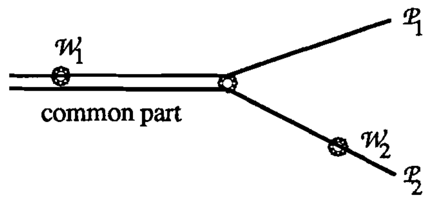
Lemma 5.2.7 The 'W→$^\mathcal{R}$W'-Functions Reach all $\Re^\mathcal{R}$-Accessible Worlds.
Given an MM-interpretation $\Im = ((\mathcal{W}, \Re), \mathcal{V}, \mathcal{W}_0)$ and the translated interpretation $\Im_{CL}$,
$\forall\ \mathcal{W}_1, \mathcal{W}_2 \in \mathcal{W}$: $\Re^\mathcal{R}(\mathcal{W}_1, \mathcal{W}_2) \Leftrightarrow \exists\ \chi \in \text{'W}\to^\mathcal{R}\text{W'}_\mathcal{C}\ \forall \mathcal{P}$ with $\mathcal{W}_1 \in \mathcal{P}$: $\chi(<\mathcal{W}_1,\mathcal{P}>)_{|W} = \mathcal{W}_2$.
Proof: "⇒" $\Re^\mathcal{R}(\mathcal{W}_1, \mathcal{W}_2)$ implies there is a path $\mathcal{P}'$ containing $\mathcal{W}_1$ and $\mathcal{W}_2$ and there is an $n \in \mathcal{N}^\mathcal{R}$ such that $\mathcal{W}_2$ is reached from $\mathcal{W}_1$ in n $\Re^\varnothing$-transitions on $\mathcal{P}'$ (lemma 5.1.7,b). According to lemma 5.2.5, for a given $\mathcal{P}$ containing $\mathcal{W}_1$ there is a $p \in \text{'W}\to\text{P'}_\mathcal{C}$ with $p(<\mathcal{W}_1,\mathcal{P}>) = <\mathcal{W}_1,\mathcal{P}'>$. Now we select $\chi := p \circ +n$ and get $\chi(<\mathcal{W}_1,\mathcal{P}>)_{|W} = \mathcal{W}_2$.
"⇐" $\chi(<\mathcal{W}_1,\mathcal{P}>) = (\chi_P \circ \chi_W)(<\mathcal{W}_1,\mathcal{P}>)$ for some $\chi_P \in \text{'W}\to\text{P'}_\mathcal{C}$ and $\chi_W \in \text{'P}\to^\mathcal{R}\text{W'}_\mathcal{C}$
With the two previous lemmas we get $\Re^\mathcal{R}(\mathcal{W}_1, \mathcal{W}_2)$. ■

Lemma 5.2.8 The 'D,W$\to^{\mathcal{R}}$W'-Functions Cover all Labeled $\mathfrak{R}^{\mathcal{R}}$-Transitions.
Given an MM-interpretation $\mathfrak{I} = ((\mathcal{W}, \mathfrak{R}), \mathcal{V}, \mathcal{W}_0)$ and the translated interpretation $\mathfrak{I}_{CL}$,
$\forall\, \mathcal{W}_1, \mathcal{W}_2 \in \mathcal{W}\!: \ell \in L_C\ (\mathcal{W}_1, \mathcal{W}_2) \in \mathfrak{R}^{\mathcal{R}}(\ell) \Leftrightarrow \exists\, \chi \in$ 'D,W$\to^{\mathcal{R}}$W'$_C$ $\forall \mathcal{P}$ with $\mathcal{W}_1 \in \mathcal{P}\!: \chi(\ell)(<\mathcal{W}_1, \mathcal{P}>)_{|W} = \mathcal{W}_2$.
Proof: $(\mathcal{W}_1, \mathcal{W}_2) \in \mathfrak{R}^{\mathcal{R}}(\ell)$ implies there is a path $\mathcal{P}'$ containing $\mathcal{W}_1$ and $\mathcal{W}_2$ and there is an n $\in \mathcal{N}^{\mathcal{R}}$ such that $\mathcal{W}_2$ is reached from $\mathcal{W}_1$ in n $\mathfrak{R}^{\emptyset}$-transitions on $\mathcal{P}'$ (lemma 5.1.7,b) and at least the last one is labeled with ℓ. As in lemma 5.2.7 we select $\chi(\ell) := p \circ +n \in$ 'W$\to^{\mathcal{R}}$W'$_C$ and get $\chi(\ell)(<\mathcal{W}_1, \mathcal{P}>)_{|W} = \mathcal{W}_2$. Since there is no further restriction on 'D,W$\to^{\mathcal{R}}$W'$_C$ $\chi \in$ 'D,W$\to^{\mathcal{R}}$W'$_C$
"$\Leftarrow$" Since $\chi \in$ 'D,W$\to^{\mathcal{R}}$W'$_C$ $(w_{|W}, \chi(\ell)(w)_{|W}) \in \mathfrak{R}^{\mathcal{R}}(\ell)$ per definition. ■

We are now going to prove the $\exists$-Quantifier Independency Lemma (def. 4.3.5) which ensures that the Skolemization in the translation from CL to OSPL is sound. Without this lemma, the translation from MM-Logic to CL would be useless.

Lemma 5.2.9 The $\exists$-Quantifier Independency Lemma
Given an MM-interpretation $\mathfrak{I} = ((\mathcal{W}, \mathfrak{R}), \mathcal{V}, \mathcal{W}_0)$ and the translated interpretation $\mathfrak{I}_{CL}$,
a) For all $w \in WP_C$ and for all $\exists$-quantified sets $\mathcal{Y} \subseteq \mathcal{Y}$ of $WP_C \to WP_C$ context access functions:
$\exists\, y \in \mathcal{Y}\!: \forall\, x_i \in$ 'W$\to^{rt}$W'$_C$ $y_i \in \mathcal{Y}$ $y(x_i(w)) = y_i(x_i(w))$.
b) For all $w \in WP_C$, $\ell \in D_C$ and for all $\exists$-quantified sets $\mathcal{Y} \subseteq$ 'D,W$\to^{rt}$W'$_C$:
$\exists\, y \in$ 'D,W$\to^{rt}$W'$_C$: $\forall x_i \in$ 'W$\to^{rt}$W'$_C$ $y_i \in \mathcal{Y}$ $y(\ell)(x_i(w)) = y_i(\ell)(x_i(w))$.
Proof: a) Case $\mathcal{Y} =$ 'W$\to$P'$_C$:
Select $y \in$ 'W$\to$P'$_C$ with $y(x_i(w)) = y_i(x_i(w))$. Since there is no restriction on 'W$\to$P'$_C$ such a selection is always possible. (Notice that we exploit that two different y_i do not map the same $x_i(w)$ to different worlds.)

Case $\mathcal{Y} =$ 'P$\to^{\mathcal{R}}$W'$_C$:
The functions $z \in$ 'P$\to^{\mathcal{R}}$W'$_C$ are restricted in such a way that whenever $z(<\mathcal{W}, \mathcal{P}_1>) = <\mathcal{W}_n, \mathcal{P}_1>$, for all paths $\mathcal{P}_2$ having at least the part until $\mathcal{W}_n$ in common, $z(<\mathcal{W}, \mathcal{P}_2>) = <\mathcal{W}_n, \mathcal{P}_2>$. Therefore the existence of $y \in$ 'P$\to^{\mathcal{R}}$W'$_C$ is not obvious. Fortunately the $\exists$-quantified sets in 'P$\to^{\mathcal{R}}$W'$_C$ are restricted in a similar way to match the restriction on the functions themselves. According to the semantics of the MM-Logic operators (see the remark after def. 5.1.5), existence of a world on a path always means that the following situation never occurs:



i.e., whenever on a path $\mathcal{P}_1$ the existence of a world $\mathcal{W}_1$ with a certain property is postulated then for all other paths containing $\mathcal{W}_1$, this world $\mathcal{W}_1$ is chosen as well. Therefore $\mathcal{Y}$ is restricted to not containing functions y_i and y_k mapping $\mathcal{W}$ to $\mathcal{W}_1$ and $\mathcal{W}_2$ in the above way. Thus, we can again choose $y \in$ 'P$\to^{\mathcal{R}}$W'$_C$ with $y(x_i(w)) = y_i(x_i(w))$.

Case $\mathcal{Y} =$ 'W$\to^{\mathcal{R}}$W'$_C$
Since for all $y_i \in$ 'W$\to^{\mathcal{R}}$W'$_C$ $y_i = y_{iP} \circ y_{iW}$ where $y_{iP} \in$ 'W$\to$P'$_C$ and $y_{iW} \in$ 'P$\to^{\mathcal{R}}$W'$_C$ we have $y_i(x_i(w)) = (y_{iP} \circ y_{iW})(x_i(w)) = y_{iW}(y_{iP}(x_i(w)))$. Exploiting the results of the two previous cases, we find a $y_P \in$ 'W$\to$P'$_C$ and a $y_W \in$ 'P$\to^{\mathcal{R}}$W'$_C$ such that $y_{iW}(y_{iP}(x_i(w)) = (y_P \circ y_W)(x_i(w))$. Hence we choose $y := y_P \circ y_W \in$ 'W$\to^{\mathcal{R}}$W'$_C$

b) The arguments are the same as in the last two cases of a). ■

Lemma 5.2.10 $\Psi_{\mathfrak{I}}(\mathfrak{I})$ **is well defined.**

For an MM-interpretation $\mathfrak{I} = ((\mathcal{W}, \mathfrak{R}), \mathcal{V}, \mathcal{W}_0)$, $\mathfrak{I}_{\mathrm{CL}} := \Psi_{\mathfrak{I}}(\mathfrak{I}) =: ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, (<\mathcal{W}_0, \mathcal{P}_0>), \mathcal{P})$ is a signature interpretation over $\Psi_\Sigma(\Sigma)$.

Proof: To show this we must check

a) whether the interpretations of the sort symbols are nonempty sets,
   and especially whether the interpretations of the functional sorts are sets of *total* functions.

b) whether all subsort relations are realized by corresponding set inclusions,

c) whether the sort declarations for the composition function symbol (def. 5.2.1,c) are realized by the composition function in $\mathcal{C}$,

d) whether the sort declarations for the application function (def. 5.2.1,c) are realized by the application function in $\mathcal{C}$, and

e) whether the interpretations of the generated constant, function and predicate symbols meet their sort declarations.

a) The interpretations of the sort symbols are nonempty sets:

$\mathrm{D}_{\mathcal{C}} = \mathrm{D}_{\mathcal{W}_0}$ where D is a domain sort: $\mathrm{D}_{\mathcal{W}_0} \neq \emptyset$. (def. 5.1.3,1)

$\mathrm{W}_{\mathcal{C}} = \mathcal{W}$. $\mathcal{W}_0 \in \mathcal{W}$

$\mathrm{WP}_{\mathcal{C}} = \{<\mathcal{W}, \mathcal{P}> \mid \mathcal{W} \in \mathrm{W}_{\mathcal{C}}, \mathcal{P} \in \mathcal{P}(\mathcal{W}_0) \text{ and } \mathcal{W} \in \mathcal{P}\}$

Since $\mathrm{W}_{\mathcal{C}} \neq \emptyset$, there is at least one tuple $<\mathcal{W}_0, \{\mathcal{W}_0 \ldots\}> \in \mathrm{WP}_{\mathcal{C}}$

$\text{‘W}{\to}\text{P’}_{\mathcal{C}} = \{p{:}\mathrm{WP}_{\mathcal{C}} \to \mathrm{WP}_{\mathcal{C}} \mid p(<\mathcal{W}, \mathcal{P}>)_{|\mathrm{W}} = \mathcal{W} \text{ and } p \text{ is total}\}$

At least the identity function is in $\text{‘W}{\to}\text{P’}_{\mathcal{C}}$

$\text{‘P}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} := \{\chi{:}\mathrm{WP}_{\mathcal{C}} \to \mathrm{WP}_{\mathcal{C}} \mid \forall w \in \mathrm{WP}_{\mathcal{C}}\ \exists n \in \mathcal{N}^{\mathcal{R}}\ \chi(w) = +n(w)$
$\quad\quad\text{and } \forall w' \text{ with } w'_{|\mathrm{W}} = w_{|\mathrm{W}} \text{ and } \chi(w)_{|\mathrm{W}} \in w'_{\approx\mathrm{P}}{:}\ \chi(w') = +n(w')\}$

The $+1$-function is in all $\text{‘P}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}}$-sets. Therefore they are not empty.

The functions are total because the $+n$-functions are total.

$\text{‘W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} = \{\chi{:}\mathrm{WP}_{\mathcal{C}} \to \mathrm{WP}_{\mathcal{C}} \mid \exists \chi_{\mathrm{P}} \in \text{‘W}{\to}\text{P’}_{\mathcal{C}}\ \exists \chi_{\mathrm{W}} \in \text{‘P}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}}\ \chi = \chi_{\mathrm{P}} \circ \chi_{\mathrm{W}}\}$

Since $\text{‘W}{\to}\text{P’}_{\mathcal{C}} \neq \emptyset$ and $\text{‘P}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} \neq \emptyset$, the composition of a least one ‘W→P’-function and one $\text{‘P}{\to}^{\mathcal{R}}\text{W’}$-function is in $\text{‘W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}}$

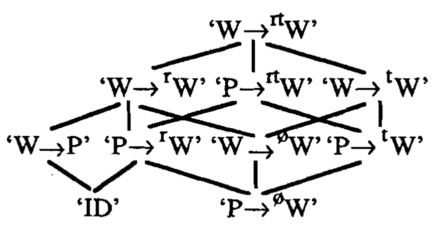
Since the components are total functions, the $\text{‘W}{\to}^{\mathcal{R}}\text{W’}$-functions are total as well.

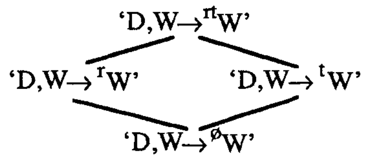
$\text{‘D,W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} = \{\chi{:}\mathrm{L}_{\mathcal{C}} \times \mathrm{WP}_{\mathcal{C}} \to \mathrm{WP}_{\mathcal{C}} \mid \forall \ell \in \mathrm{L}_{\mathcal{C}}\ \chi(\ell) \in \text{‘W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} \text{ and } \forall w \in \mathrm{WP}_{\mathcal{C}}\ (w_{|\mathrm{W}}, \chi(\ell)(w)_{|\mathrm{W}}) \in \mathfrak{R}^{\mathcal{R}}(\ell)\}$

$\text{‘W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} \neq \emptyset$ and the seriality of labeled transitions (def. 5.1.3,2e) implies $\text{‘D,W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} \neq \emptyset$ and the functions are total.

b) All subsort relations are realized by corresponding set inclusions.

The subsort relationships are:





$\text{‘ID’}_{\mathcal{C}} \subseteq \text{‘W}{\to}\text{P’}_{\mathcal{C}}$:

Since $\text{‘ID’}_{\mathcal{C}} = \{\mathrm{ID}_{\mathcal{C}}\}$ and $\mathrm{ID}_{\mathcal{C}}(<\mathcal{W}, \mathcal{P}>)_{\mathrm{W}} = \mathcal{W}$, $\mathrm{ID}_{\mathcal{C}} \in \text{‘W}{\to}\text{P’}_{\mathcal{C}}$, i.e. $\text{‘ID’}_{\mathcal{C}} \subseteq \text{‘W}{\to}\text{P’}_{\mathcal{C}}$

$\text{‘ID’}_{\mathcal{C}} \subseteq \text{‘P}{\to}^{\mathrm{r}}\text{W’}_{\mathcal{C}}$:

$\mathrm{ID}_{\mathcal{C}} = +0 \in \text{‘P}{\to}^{\mathrm{r}}\text{W’}_{\mathcal{C}}$.

$\text{‘W}{\to}\text{P’}_{\mathcal{C}} \subseteq \text{‘W}{\to}^{\mathrm{r}}\text{W’}_{\mathcal{C}}$:

Let $\chi \in \text{‘W}{\to}\text{P’}_{\mathcal{C}}$. Since $\chi = \chi \circ \mathrm{ID}_{\mathcal{C}}$ and $\mathrm{ID}_{\mathcal{C}} \in \text{‘P}{\to}^{\mathrm{r}}\text{W’}_{\mathcal{C}}$, $\chi \in \text{‘W}{\to}^{\mathrm{r}}\text{W’}_{\mathcal{C}}$ and therefore $\text{‘W}{\to}\text{P’}_{\mathcal{C}} \subseteq \text{‘W}{\to}^{\mathrm{r}}\text{W’}_{\mathcal{C}}$.

$\text{‘P}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}} \subseteq \text{‘W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}}$, $\mathcal{R} \in \{\emptyset, \mathrm{r}, \mathrm{t}, \mathrm{rt}\}$:

Let $\chi \in \text{‘P}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}}$. Since $\chi = \mathrm{ID}_{\mathcal{C}} \circ \chi$ and $\mathrm{ID}_{\mathcal{C}} \in \text{‘W}{\to}\text{P’}_{\mathcal{C}}$, $\chi \in \text{‘W}{\to}^{\mathcal{R}}\text{W’}_{\mathcal{C}}$ and therefore

'P→$^{\mathcal{R}}$W'$_{C}$ ⊆ 'W→$^{\mathcal{R}}$W'$_{C}$

'P→^{p}W'$_{C}$ ⊆ 'P→^{q}W'$_{C}$ according to the above sort lattice.

The subset relationships follow trivially from the definition of the 'P→$^{\mathcal{R}}$W'-functions and the corresponding subset relationships of $\mathcal{N}^{\mathcal{R}}$ (lemma 5.1.7,b).

'W→^{p}W'$_{C}$ ⊆ 'W→^{q}W'$_{C}$ p and q according to the sort lattice.

Let $\chi \in$ 'W→^{p}W'$_{C}$ i.e. $\exists \chi_P \in$ 'W→P' $\exists \chi_W \in$ 'P→^{p}W'$_{C}$ $\chi = \chi_P \circ \chi_W$.

Since 'P→^{p}W'$_{C}$ ⊆ 'P→^{q}W'$_{C}$ (see the above cases), $\chi_W \in$ 'P→^{q}W'$_{C}$

Hence, $\chi \in$ 'W→^{q}W'$_{C}$ and consequently 'W→^{p}W'$_{C}$ ⊆ 'W→^{q}W'$_{C}$

'D,W→^{p}W'$_{C}$ ⊆ 'D,W→^{q}W'$_{C}$ p and q according to the sort lattice.

Let $\chi \in$ 'D,W→^{p}W'$_{C}$ i.e. $\forall \ell \in D_C$ $\chi(\ell) \in$ 'W→^{p}W'$_{C}$.

Since 'W→^{p}W'$_{C}$ ⊆ 'W→^{q}W'$_{C}$ (see the above cases),$\forall \ell \in D_C \chi(\ell) \in$ 'W→^{q}W'$_{C}$

Hence, $\chi \in$ 'D,W→^{q}W'$_{C}$ and consequently 'D,W→^{p}W'$_{C}$ ⊆ 'D,W→^{q}W'$_{C}$

c) The sort declarations for the composition function symbol are realized by the composition function in $\mathcal{C}$.
The sort declarations are (def. 5.2.1,2c):

∘: 'W→^{p}W' × 'W→^{q}W' → 'W→^{s}W'	and ∘:	'W→^{p}W' × 'W→P' → 'W→^{p}W'
'P→^{p}W' × 'P→^{q}W' → 'P→^{s}W'		'W→P' × 'W→^{q}W' → 'W→^{q}W'
'D,W→^{p}W' × 'D,W→^{q}W' → 'D,W→^{s}W'		'W→P' × 'W→P' → 'W→P'

where s is derived from p and q with the following matrix:

p \ q	∅	r	t	rt
∅	t	t	t	t
r	t	rt	t	rt
t	t	t	t	t
rt	t	rt	t	rt

We begin with 'P→^{p}W' × 'P→^{q}W' → 'P→^{s}W':

Let $\chi \in$ 'P→^{p}W'$_{C}$, $y \in$ 'P→^{q}W'$_{C}$ and $w \in \mathrm{WP}_C$

$(\chi \circ y)(w) = (+n \circ +m)(w) = +(n+m)(w)$ for some n ∈ $\mathcal{N}^{p}$ and m ∈ $\mathcal{N}^{q}$.

The different cases for p,q and s are now trivially to be verified.

'W→P' × 'W→P' → 'W→P':

Let $p \in$ 'W→P'$_{C}$, $q \in$ 'W→P'$_{C}$ and $<\mathcal{W},\mathcal{P}> \in \mathrm{WP}_C$

$(p \circ q)\ (<\mathcal{W},\mathcal{P}>)\ = q\,(<\mathcal{W},\mathcal{P}_1>)$

$= <\mathcal{W},\mathcal{P}_2>$.

Thus, $p \circ q \in$ 'W→P'$_{C}$.

'P→$^{\mathcal{R}}$W' × 'W→P' → 'W→$^{\mathcal{R}}$W':

Let $\chi \in$ 'P→^{p}W'$_{C}$, $p \in$ 'W→P'$_{C}$, $w = <\mathcal{W}_n,\mathcal{P}> \in \mathrm{WP}_C$.

$(\chi \circ p)(<\mathcal{W}_n,\mathcal{P}>) = p(<\mathcal{W}_{n+m},\mathcal{P}>)$ for some m ≥ 0

$= <\mathcal{W}_{n+m},\mathcal{P}'>$ for some path $\mathcal{P}'$

$\Rightarrow \mathcal{W}_n \in \mathcal{P}'$

Therefore we can define $q(<\mathcal{W},\mathcal{P}>) := <\mathcal{W},p(\chi(<\mathcal{W},\mathcal{P}>))_{|P}>$, $q \in$ 'P→^{p}W'$_{C}$

We have $\chi(q(<\mathcal{W},\mathcal{P}>)) = \chi(<\mathcal{W},p(\chi(<\mathcal{W},\mathcal{P}>))_{|P}>)$

$= \chi(p(<\mathcal{W},\chi(<\mathcal{W},\mathcal{P}>)_{|P}>))$ (p doesn't change worlds)

$= \chi(p(<\mathcal{W},\mathcal{P}>))$ (χ does not change paths)

$\Rightarrow \chi \circ p = q \circ \chi$

$\Rightarrow \chi \circ p \in$ 'W→$^{\mathcal{R}}$W'

‘W$\to^{\mathcal{R}}$W’ $\times$ ‘W$\to$P’ $\to$ ‘W$\to^{\mathcal{R}}$W’:

Let $\chi \in$ ‘W$\to^{p}$W’$_{\mathcal{C}}$ and $p \in$ ‘W$\to$P’$_{\mathcal{C}}$.

$\chi \circ p = \chi_P \circ \chi_W \circ p$ $\qquad \chi_W \in$ ‘P$\to^{\mathcal{R}}$W’

$\quad = \chi_P \circ q \circ \chi_W$ $\qquad$ (above case)

$\quad = p' \circ \chi_W$ $\qquad$ ($p' \in$ ‘W$\to$P’)

$\Rightarrow \chi \circ p \in$ ‘W$\to^{\mathcal{R}}$W’.

‘W$\to^{p}$W’ $\times$ ‘W$\to^{q}$W’ $\to$ ‘W$\to^{s}$W’:

Let $\chi \in$ ‘W$\to^{p}$W’$_{\mathcal{C}}$ and $y \in$ ‘W$\to^{q}$W’$_{\mathcal{C}}$

$\chi \circ y = \chi_P \circ \chi_W \circ y_P \circ y_W$ $\quad$ (def. of ‘W$\to^{p}$W’$_{\mathcal{C}}$, $\chi_W \in$ ‘P$\to^{p}$W’$_{\mathcal{C}}$, $y_P \in$ ‘W$\to$P’$_{\mathcal{C}}$)

$\quad = \chi_P \circ p \circ \chi_W \circ y_W$ $\quad$ (above case, $p \in$ ‘W$\to$P’)

$\quad = z_P \circ z_W$ $\quad$ (for some $z \in$ ‘W$\to^{s}$W’$_{\mathcal{C}}$ see above case for $z_W \in$ ‘P$\to^{s}$W’$_{\mathcal{C}}$)

$\quad = z \in$ ‘W$\to^{s}$W’$_{\mathcal{C}}$

‘W$\to$P’ $\times$ ‘W$\to^{\mathcal{R}}$W’ $\to$ ‘W$\to^{\mathcal{R}}$W’:

Let $p \in$ ‘W$\to$P’$_{\mathcal{C}}$ and $y \in$ ‘W$\to^{\mathcal{R}}$W’$_{\mathcal{C}}$ and $<\mathcal{W},\mathcal{P}> \in \mathrm{WP}_{\mathcal{C}}$

$p \circ \chi = p \circ (\chi_P \circ \chi_W)$

$\quad = (p \circ \chi_P) \circ \chi_W$, $\; p \circ \chi_P \in$ ‘W$\to$P’$_{\mathcal{C}}$

$\Rightarrow p \circ \chi \in$ ‘W$\to^{\mathcal{R}}$W’$_{\mathcal{C}}$.

‘D,W$\to^{p}$W’ $\times$ ‘D,W$\to^{q}$W’ $\to$ ‘D,W$\to^{s}$W’

Let $\chi \in$ ‘D,W$\to^{p}$W’$_{\mathcal{C}}$, $y \in$ ‘D,W$\to^{q}$W’$_{\mathcal{C}}$ and $<\mathcal{W},\mathcal{P}> \in \mathrm{WP}_{\mathcal{C}}$

i.e. $\forall \ell \in \mathrm{D}_{\mathcal{C}}$ $\chi(\ell) \in$ ‘W$\to^{p}$W’$_{\mathcal{C}}$ and $y(\ell) \in$ ‘W$\to^{q}$W’$_{\mathcal{C}}$

$\Rightarrow \forall \ell \in \mathrm{D}$ $\chi(\ell) \circ y(\ell) = (\chi \circ y)(\ell) \in$ ‘W$\to^{s}$W’$_{\mathcal{C}}$ $\qquad$ (see above and def. 3.4.1,5).

i.e. $\chi \circ y \in$ ‘D,W$\to^{s}$W’$_{\mathcal{C}}$

d) The sort declarations for the application function are realized by the application function in $\mathcal{C}$.

The sort declarations for the application function $\downarrow$ are:

$\downarrow$: ‘X$\to^{\mathcal{R}}$Y’ $\times$ WP $\to$ WP $\quad$ for X,Y $\in$ {W, P}

‘D,W$\to^{\mathcal{R}}$W’ $\times$ L $\to$ ‘W$\to^{\mathcal{R}}$W’

In both cases the statement follows immediately from the semantic definition of the context sorts.

e) The interpretations of the generated constant, function and predicate symbols meet their sort declarations.

Constant symbols:

ID: ‘ID’

ID$_{\mathcal{C}}$ is the identity function on WP$_{\mathcal{C}}\to$WP$_{\mathcal{C}}$ i.e. ID$_{\mathcal{C}} \in$ ‘ID’$_{\mathcal{C}}$

IDL:‘D,W$\to^{r}$W’

IDL$_{\mathcal{C}}$ is the identity function on D$_{\mathcal{C}}\times$WP$_{\mathcal{C}}\to$WP$_{\mathcal{C}}$.

Since the reflexive transitions in $\mathfrak{R}^{r}$ are labeled with all labels (def. 5.1.3,2f),

IDL$_{\mathcal{C}} \in$ ‘D,W$\to^{r}$W’$_{\mathcal{C}}$

+1: ‘P$\to^{\emptyset}$W’

+1$_{\mathcal{C}}$ = $+1 \in$ ‘P$\to^{\emptyset}$W’$_{\mathcal{C}}$

+1L: ‘D,W$\to^{\emptyset}$W’

+1L$_{\mathcal{C}}$ = $+1L \in$ ‘D,W$\to^{\emptyset}$W’$_{\mathcal{C}}$ per definition.

Function symbols: In particular we have to show that the function symbols are interpreted as total functions.

PW: WP $\to$ W

“$\forall <\mathcal{W},\mathcal{P}> \in \mathrm{WP}_{\mathcal{C}}$ PW$_{\mathcal{C}}$($<\mathcal{W},\mathcal{P}>$) = $\mathcal{W}$”. Obviously PW$_{\mathcal{C}}$($<\mathcal{W},\mathcal{P}>$) $\in$ W$_{\mathcal{C}}$

PA: WP $\to$ ‘W$\to$P’

“$\forall w \in \mathrm{WP}_{\mathcal{C}}$ PA$_{\mathcal{C}}(w) = p$ where $p \in$ ‘W$\to$P’$_{\mathcal{C}}$ and $\forall w' \in \mathrm{WP}_{\mathcal{C}}$

if $w'_{|W} = w_{|W}$ then $p(w') = w$ else $p(w') = w'$”

Obviously PA$_{\mathcal{C}}(w) \in$ ‘W$\to$P’$_{\mathcal{C}}$ Since there are no restrictions on ‘W$\to$P’$_{\mathcal{C}}$, PA$_{\mathcal{C}}(w)$ always exists.

BF: D $\rightarrow$ ‘P$\rightarrow^{rt}$W’

“$\forall \ell \in D_{\mathcal{C}}$ BF(ℓ) = χ where $\chi \in$ ‘P$\rightarrow^{rt}$W’$_{\mathcal{C}}$ and $\forall w \in WP_{\mathcal{C}}$ $(\chi(w)_W, (\chi \circ +1)(w))_W) \in \mathfrak{R}^{\emptyset}(\ell)$

and $\chi(w)_W$ is the first world among the $y(w)_W$ with $(y(w)_W, (y \circ +1)(w))_W) \in \mathfrak{R}^{\emptyset}(\ell)$.”

Because of the fairness condition on paths (def. 5.1.3,2g), there exists for each label ℓ on each path an ℓ-labeled transition. Therefore the function $\chi \in$ ‘P$\rightarrow^{rt}$W’$_{\mathcal{C}}$ jumping to the world before that transition, exists, i.e. $\forall \ell \in D_{\mathcal{C}}$ BF$_{\mathcal{C}}(\ell) \in$ ‘P$\rightarrow^{rt}$W’$_{\mathcal{C}}$

MP: ‘W$\rightarrow^{rt}$W’ $\rightarrow$ ‘W$\rightarrow$P’

MW: ‘W$\rightarrow^{\mathcal{R}}$W’ $\rightarrow$ ‘P$\rightarrow^{\mathcal{R}}$W’

Since each ‘W$\rightarrow^{\mathcal{R}}$W’-function χ is a composition of a ‘W$\rightarrow$P’-function MP(χ) and a ‘P$\rightarrow^{\mathcal{R}}$W’-function MW($\chi$), the proof is trivial.

LT: ‘W$\rightarrow^{p}$W’ $\times$ ‘D,W$\rightarrow^{q}$W’ $\rightarrow$ ‘D,W$\rightarrow^{s}$W’

Since LT$_{\mathcal{C}}$ works similar to the composition function, the proof for $\circ$ carries over to LT.

-R: ‘W$\rightarrow^{rt}$W’ $\rightarrow$ ‘W$\rightarrow^{t}$W’   ‘P$\rightarrow^{rt}$W’ $\rightarrow$ ‘P$\rightarrow^{t}$W’

‘W$\rightarrow^{r}$W’ $\rightarrow$ ‘W$\rightarrow^{\emptyset}$W’   ‘P$\rightarrow^{r}$W’ $\rightarrow$ ‘P$\rightarrow^{\emptyset}$W’

“$\forall \chi \in$ ‘X$\rightarrow^{ry}$W’$_{\mathcal{C}}$ where X $\in$ {W, P} and $^{y} \in$ {"", ^{t}}

-R$_{\mathcal{C}}(\chi) = y$ where $y \in$ ‘X$\rightarrow^{y}$W’$_{\mathcal{C}}$ and $\forall w \in WP_{\mathcal{C}}$ $\chi(w) \neq w \Rightarrow \chi(w) = y(w)$”

Since -R$_{\mathcal{C}}(\chi)$ is allowed to map those w where χ operates as the identity to some appropriate w', -R$_{\mathcal{C}}(\chi)$ always exists and is of the right type.

FS: ‘P$\rightarrow^{t}$W’$\rightarrow$ ‘P$\rightarrow^{t}$W’

‘P$\rightarrow^{rt}$W’$\rightarrow$ ‘P$\rightarrow^{rt}$W’

“$\forall \chi \in$ ‘P$\rightarrow^{t}$W’$_{\mathcal{C}}$ $\forall w \in WP_{\mathcal{C}}$

if $\chi(w) = +(n+1)(w)$ for some n>0 then FS$_{\mathcal{C}}(\chi)(w) = +n(w)$ else FS$_{\mathcal{C}}(\chi)(w) = +1(w)$

$\forall \chi \in$ ‘P$\rightarrow^{rt}$W’$_{\mathcal{C}}$ \ ‘P$\rightarrow^{t}$W’$_{\mathcal{C}}$ $\forall w \in WP_{\mathcal{C}}$

if $\chi(w) = +(n+1)(w)$ for some n≥0 then FS$_{\mathcal{C}}(\chi)(w) = +n(w)$ else FS$_{\mathcal{C}}(\chi)(w) = w$”

Obviously FS$_{\mathcal{C}}(\chi)$ exists and has the right type.

$\rightarrowtail$: ‘P$\rightarrow^{rt}$W’ $\times$ ‘P$\rightarrow^{rt}$W’ $\rightarrow$ ‘P$\rightarrow^{rt}$W’

“$\forall \chi, y \in$ ‘P$\rightarrow^{rt}$W’$_{\mathcal{C}}$ $\forall w \in WP_{\mathcal{C}}$

if $\chi(w) = +n(w)$ for some n and

$y(w) = +m(w)$ for some m ≥ n then $\rightarrowtail_{\mathcal{C}}(\chi, y)(w) = +(m-n)(w)$ else $\rightarrowtail_{\mathcal{C}}(\chi, y)(w) = w$”

Obviously $\rightarrowtail_{\mathcal{C}}(\chi, y) \in$ ‘P$\rightarrow^{rt}$W’$_{\mathcal{C}}$ always exists.

Predicate symbols

≤: WP×WP

“$\leq_{\mathcal{C}} := \{(w_1, w_2) \mid w_1, w_2 \in WP_{\mathcal{C}}$ and $w_2 = +n(w_1)$ for some $n \geq 0\}$”

<: WP×WP

“$<_{\mathcal{C}} := \{(w_1, w_2) \mid w_1, w_2 \in WP_{\mathcal{C}}$ and $w_2 = +n(w_1)$ for some $n > 0\}$”

Its obvious that the sort declarations meet the corresponding semantic definition. ■

Lemma 5.2.11 The Axiomatization of the Possible Worlds Structure is Satisfied

For every MM-interpretation $\mathfrak{I}$, the axioms generated by the specification morphism $\Psi_{\mathcal{S}}$ (def. 5.2.3,A1-J3) are satisfied by the translated MM-interpretation $\Psi_{\mathfrak{I}}(\mathfrak{I})$.

Proof: We check the axioms one by one.

Characterization of $\circ$ and $\downarrow$:

A1 ∀x,y:‘W$\rightarrow^{rt}$W’ ∀w:WP $\downarrow$(x, w) = $\downarrow$(y, w) $\Rightarrow$ x = y

A2 ∀x,y:‘D,W$\rightarrow^{rt}$W’ ∀l:D ∀w:WP $\downarrow$($\downarrow$(x, l), w) = $\downarrow$($\downarrow$(y, l), w) $\Rightarrow$ x = y

A3 ∀x,y:‘W$\rightarrow^{rt}$W’ ∀w:WP $\downarrow$(x $\circ$ y, w) = $\downarrow$(y, $\downarrow$(x, w))

A4 ∀x,y:‘D,W$\rightarrow^{rt}$W’ ∀l:D $\downarrow$(x $\circ$ y, l) = $\downarrow$(x, l) $\circ$ $\downarrow$(y, l)

These axioms are satisfied because $\mathfrak{I}_{CL}$ is a functional interpretation (theorem 3.4.3). The axiom A4 holds because, since only the last label matters in a sequence of transitions, the composition of two ℓ-labeled transitions is again ℓ-labeled.

Identity functions.

B1 $\forall$w:WP $\downarrow$(ID, w) = w   Trivial

B2 $\forall$w:WP $\forall$l:D $\downarrow$($\downarrow$(IDL, l), w) = w

B2 is satisfied because the reflexive transitions are labeled with all labels.

Characterization of the 'W→P'-functions

C1 $\forall$p:'W→P'   MP(p) = p

C2 $\forall$p:'W→P'   MW(p) = ID

C3 $\forall$ w_1, w_2:WP $PW(w_1) = PW(w_2) \Rightarrow w_2 = PA(w_2)(w_1)$

Satisfiability is checked straightforwardly from the semantics of MP, MW and PA.

Characterization of the 'P→$^{\mathcal{R}}$W'-functions.

D1 $\forall$x:'P→^{rt}W'   MW(x) = x

D2 $\forall$x:'P→^{rt}W'   MP(x) = ID

Obvious.

D3 $\forall$x,y:'P→^{rt}W' $\forall$p:'W→P' $\forall$w:WP $x(w) = y(w) \Rightarrow PW(MP(x \circ z \circ p) \circ y)(w)) = PW(x(w))$

Let $x, y \in$ 'P→^{rt}W'$_{\mathcal{C}}$ $p \in$ 'W→P'$_{\mathcal{C}}$ $w = <\mathcal{W},\mathcal{P}> \in WP_{\mathcal{C}}$ with $x(w) = y(w) = +n(w) =: <\mathcal{W}_n,\mathcal{P}>$.

Let $w' := MP_{\mathcal{C}}(x \circ z \circ p))(w) = MP_{\mathcal{C}}(x \circ z \circ p))(<\mathcal{W},\mathcal{P}>)$.

$= <\mathcal{W}, (x \circ z \circ p)(<\mathcal{W},\mathcal{P}>)_{|P}>$ (def. of $MP_{\mathcal{C}}$, 5.2.4)

$= <\mathcal{W}, (z \circ p)(<\mathcal{W}_n,\mathcal{P}>)_{|P}> = <\mathcal{W}, p(<\mathcal{W}_{n+m},\mathcal{P}>)_{|P}>$ for some m

$= <\mathcal{W}, <\mathcal{W}_{n+m},\mathcal{P}'>_{|P}>$ for some $\mathcal{P}'$.

$\Rightarrow$ $x(w)_{|W} = y(w)_{|W} = \mathcal{W}_n \in w'_{\approx P}$

$\Rightarrow$ $x(w') = y(w') = +n(w)$ (def. of 'P→$^{\mathcal{R}}$W'$_{\mathcal{C}}$, 5.2.4)

$\Rightarrow$ $(MP_{\mathcal{C}}(x \circ z \circ p) \circ y))(w)_{|W} = x(w)_{|W}$.

D4 $\forall$x,y,z:'P→^{rt}W' $\forall$p:'W→P' $\forall$w:WP $x(w) = y(w) \Leftarrow PW(MP(x \circ z \circ p) \circ y)(w)) = PW(x(w))$

Let $x, y \in$ 'P→^{rt}W'$_{\mathcal{C}}$ $p \in$ 'W→P'$_{\mathcal{C}}$ $w = <\mathcal{W},\mathcal{P}> \in WP_{\mathcal{C}}$

with $(MP_{\mathcal{C}}(x \circ z \circ p) \circ y))(w)_{|W} = x(w)_{|W} = +n(w)_{|W} =: \mathcal{W}_n$

$\Rightarrow$ $MP_{\mathcal{C}}(x \circ z \circ p) \circ y))(w)_{|W} = \mathcal{W}_n \in w_{\approx P}$

$\Rightarrow$ $y(w) = +n(w) = x(w)$.

D5 The injectivity of $+1$ holds because of the isomorphism of paths to natural numbers.

Characterization of the 'W→$^{\mathcal{R}}$W'-functions.

E1 $\forall$x:'W→^{rt}W' x = MP(x) $\circ$ MW(x)

Let $x \in$ 'W→^{rt}W'$_{\mathcal{C}}$, $w = <\mathcal{W},\mathcal{P}> \in WP_{\mathcal{C}}$

$MP_{\mathcal{C}}(x) \circ MW_{\mathcal{C}}(x)(<\mathcal{W},\mathcal{P}>)$

$= MW_{\mathcal{C}}(x)(<\mathcal{W}, x(<\mathcal{W},\mathcal{P}>)_{|P}>)$

$= <x(<\mathcal{W},\mathcal{P}>)_{|W}, x(<\mathcal{W},\mathcal{P}>)_{|P}>$

$= x(<\mathcal{W},\mathcal{P}>)$.

E2 $\forall$x,y:'W→^{rt}W' $\forall$w:WP $x(w) = y(w) \Rightarrow MP(x)(w) = MP(y)(w)$

Let $x, y \in$ 'W→^{rt}W'$_{\mathcal{C}}$, $w = <\mathcal{W},\mathcal{P}> \in WP_{\mathcal{C}}$ and $x(w) = y(w)$

$MP_{\mathcal{C}}(x)(w) = <\mathcal{W}, x(w)_{|P}> = <\mathcal{W}, y(w)_{|P}> = MP_{\mathcal{C}}(y)(w)$.

E3 $\forall$x,y:'W→^{rt}W' $\forall$w:WP $x(w) = y(w) \Rightarrow MW(x)(w) = MW(y)(w)$

Let $x, y \in$ 'W→^{rt}W'$_{\mathcal{C}}$, $w = <\mathcal{W},\mathcal{P}> \in WP_{\mathcal{C}}$ and $x(w) = y(w)$

Case $w = <\mathcal{W}, (x \circ z)(w)_{|P}>$ for some $z \in$ 'W→^{rt}W'$_{\mathcal{C}}$

$\Rightarrow$ $MW_{\mathcal{C}}(x)(w) = <x(w)_{|W}, (x \circ z)(w)_{|P}> = <y(w)_{|W}, (y \circ z)(w)_{|P}> = MP_{\mathcal{C}}(y)(w)$.

Case $w \neq \langle \mathcal{W}, (\chi \circ z))(w)_{|P}\rangle$

$\Rightarrow$ $MW_{\mathcal{C}}(\chi)(w) = \langle \chi(w)_{|W}, \mathcal{P}\rangle = \langle y(w)_{|W}, \mathcal{P}\rangle = MP_{\mathcal{C}}(y)(w)$

E4 $\forall x,y$:'W$\rightarrow^{rt}$W' $\qquad MW(x \circ y) = MW(x) \circ MW(y)$

Let $\chi, y \in$ 'W$\rightarrow^{rt}$W'$_{\mathcal{C}}$ and $w = \langle \mathcal{W}, \mathcal{P}\rangle \in WP_{\mathcal{C}}$

Case $w = \langle \mathcal{W}, (\chi \circ y \circ z)(w)_{|P}\rangle$ for some $z \in$ 'W$\rightarrow^{rt}$W'$_{\mathcal{C}}$

$\Rightarrow$ $MW_{\mathcal{C}}(\chi \circ y)(w) = \langle (\chi \circ y)(w)_{|W}, (\chi \circ y \circ z)(w)_{|P}\rangle = MW_{\mathcal{C}}(\chi)(\langle y(w)_{|W}, (\chi \circ y \circ z)(w)_{|P}\rangle)$

$= (MW_{\mathcal{C}}(\chi) \circ MW_{\mathcal{C}}(y))(\langle \mathcal{W}, (\chi \circ y \circ z)(w)_{|P}\rangle)$

$= (MW_{\mathcal{C}}(\chi) \circ MW_{\mathcal{C}}(y))(w)$

Case $w \neq \langle \mathcal{W}, (\chi \circ y \circ z)(w)_{|P}\rangle$

$\Rightarrow$ $MW_{\mathcal{C}}(\chi \circ y)(w) = \langle (\chi \circ y)(w)_{|W}, \mathcal{P}\rangle = MW_{\mathcal{C}}(\chi)(\langle y(w)_{|W}, \mathcal{P}\rangle)$

$= (MW_{\mathcal{C}}(\chi) \circ MW_{\mathcal{C}}(y))(\langle \mathcal{W}, \mathcal{P}\rangle)$

$= (MW_{\mathcal{C}}(\chi) \circ MW_{\mathcal{C}}(y))(w).$

The satisfiability proofs for the remaining axioms are straightforward. ■

Lemma 5.2.12 Soundness Lemma for the Translation into CL

If $\mathcal{S}$ is an MM-specification satisfied by $\mathfrak{I} = ((\mathcal{W}, \mathfrak{R}), \mathcal{V}, \mathcal{W})$ then the formulae in the translated specification are satisfied by the translated model $\Psi_{\mathfrak{I}}(\mathfrak{I}) = \mathfrak{I}_{CL} =: ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, (\langle \mathcal{W}, \mathcal{P}\rangle), \mathcal{P})$ where $\mathcal{P}$ may be any path and $\mathcal{P}$ is arbitrary.

Proof: We show this by induction on the structure of MM-formulae:

Base Case: The atomic level is the base case.

The translated atoms equal the original ones. The actual structure in which an atom A is interpreted is $PW_{\mathcal{C}}(\langle \mathcal{W}, \mathcal{P}\rangle) = \mathcal{W}$. Therefore $\mathfrak{I}_{CL}$ satisfies A as well.

Induction Step: Let $\mathcal{G}$ be a non atomic formula.

The precondition is always $\mathfrak{I} = ((\mathcal{W}, \mathfrak{R}), \mathcal{V}, \mathcal{W}) \models_M \mathcal{G}$.

The induction hypothesis is: For every subformula $\mathcal{F}$ of $\mathcal{G}$: If $\mathfrak{I}' \models_M \mathcal{F}$ then $\mathfrak{I}'_{CL} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ for every path $\mathcal{P}'$.

Let $w := \langle \mathcal{W}, \mathcal{P}\rangle$. For convenience $PW_{\mathcal{C}}(\mathcal{X})$ is again abbreviated as $\mathcal{X}_{|W}$.

In the sequel we frequently apply the results of the lemmas 5.2.5 to 5.2.9 about the correspondences between the context access functions and the accessibility relations.

We perform a case analysis according to the structure of $\mathcal{F}$. The interesting cases are the modal operators:

Case $\mathcal{G} = \Box^{\mathcal{R}} \mathcal{F}$

The translation rule is $\Psi_{\mathcal{F}}(\Box^{\mathcal{R}} \mathcal{F}) = \forall x$:'W$\rightarrow^{\mathcal{R}}$W' $\Psi_{\mathcal{F}}(\mathcal{F})$

Let $\chi \in$ 'W$\rightarrow^{\mathcal{R}}W'_{\mathcal{C}}$.

Since $\mathfrak{I}[\chi(w)_{|W}] \models_M \mathcal{F}$ (def. 5.1.5) we can apply the induction hypothesis and obtain $\mathfrak{I}_{CL}[WP/\chi(w)]_{\mathcal{C}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ and since $x \notin \Psi_{\mathcal{F}}(\mathcal{F})$, $\mathfrak{I}_{CL}[x/\chi]_{\mathcal{V}}[WP/\chi(w)]_{\mathcal{C}}[x/w]_{\mathcal{P}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$.

Def. 4.2.5 finally gives us $\mathfrak{I}_{CL} \models_C \Psi_{\mathcal{F}}(\mathcal{G})$.

Case $\mathcal{G} = [\![t]\!]^{\mathcal{R}} \mathcal{F}$

The translation rule is $\Psi_{\mathcal{F}}([\![t]\!]^{\mathcal{R}} \mathcal{F}) = \forall\downarrow$(x:'D,W$\rightarrow^{\mathcal{R}}$W', z:S(t)=t) $\Psi_{\mathcal{F}}(\mathcal{F})$.

Let $\chi \in$ 'D,W$\rightarrow^{\mathcal{R}}W'_{\mathcal{C}}$. Since $\mathfrak{I}[\chi(\mathfrak{I}(t), w)_{|W}] \models_M \mathcal{F}$ we can apply the induction hypothesis and obtain $\mathfrak{I}_{CL}[WP/\chi(\mathfrak{I}(t), w)]_{\mathcal{C}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ and since $x \notin \Psi_{\mathcal{F}}(\mathcal{F})$ and $z \notin \Psi_{\mathcal{F}}(\mathcal{F})$, $\mathfrak{I}_{CL}[x/\chi]_{\mathcal{V}}[WP/\chi(\mathfrak{I}(t), w)]_{\mathcal{C}}[x/w]_{\mathcal{P}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$. Def. 4.2.5 finally gives us $\mathfrak{I}_{CL} \models_C \Psi_{\mathcal{F}}(\mathcal{G})$.

Case $\mathcal{G} = \blacktriangleright \mathcal{F}$

The translation rule is $\Psi_{\mathcal{F}}(\blacktriangleright \mathcal{F}) = \forall p$:'W$\rightarrow$P' $\exists x$:'P$\rightarrow^{rt}$W' $\Psi_{\mathcal{F}}(\mathcal{F})$.

Let $p \in$ 'W$\rightarrow$P'$_{\mathcal{C}}$. There is a $\chi \in$ 'P$\rightarrow^{rt}$W' with $\mathfrak{I}[\chi(p(w))_{|W}] \models_M \mathcal{F}$. (def. 5.1.5)

We apply the induction hypothesis obtaining $\mathfrak{I}_{CL}[WP/\chi(p(w))]_{\mathcal{C}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ and since $p \notin \Psi_{\mathcal{F}}(\mathcal{F})$ and $z \notin \Psi_{\mathcal{F}}(\mathcal{F})$, $\mathfrak{I}_{CL}[p/p, x/\chi]_{\mathcal{V}}[WP/\chi(p(w))]_{\mathcal{C}}[p/w, x/p(w)]_{\mathcal{P}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$.

Thus, $\mathfrak{I}_{CL}[p/p]_{\mathcal{V}}[WP/p(w)]_{\mathcal{C}}[p/w]_{\mathcal{P}} \models_C \exists x$:'P$\rightarrow^{rt}$W' $\Psi_{\mathcal{F}}(\mathcal{F})$. (def. 4.2.5)

and finally $\mathfrak{I}_{CL} \models_C \Psi_{\mathcal{F}}(\mathcal{G})$. (def. 4.2.5)

Case $\mathcal{G} = |t)\mathcal{F}$

The translation rule is $\Psi_{\mathcal{J}}(|t)\mathcal{F}) = \forall p{:}\text{'W}\rightarrow\text{P'}\ \wp(\mathrm{BF}(t) \circ +1)\ \Psi_{\mathcal{J}}(\mathcal{F})$.

Let $p \in \text{'W}\rightarrow\text{P'}_{\mathcal{C}}$, $\mathfrak{I}_{CL}' := \mathfrak{I}_{CL}[p/p]_{\mathcal{V}}[WP/p(w)]_{\mathcal{C}}[p/w]_{\mathcal{P}}$ and $p(w) =: \langle\mathcal{W},\mathcal{P}\rangle$

Since $\mathfrak{I} \models_M \mathcal{G}$, $\mathfrak{I}[\mathcal{W}_1] \models_M \mathcal{F}$ where $\mathcal{W}_1$ is the first world on $\mathcal{P}$ after an $\mathfrak{I}(t)$-labeled transition. According to the semantic definition of BF and $+1$, (def. 5.2.4), for $w_1 := \mathfrak{I}_{CL}'(\mathrm{BF}(t) \circ +1)(w)$, exactly $w_{1|W} = \mathcal{W}_1$.

Therefore, applying the induction hypothesis, we find $\mathfrak{I}_{CL}'[WP/w_1]_{\mathcal{C}} \models_C \Psi_{\mathcal{J}}(\mathcal{F})$.

Thus, $\mathfrak{I}_{CL}' \models_C \wp(\mathrm{BF}(t) \circ +1)\ \Psi_{\mathcal{J}}(\mathcal{F})$ and finally $\mathfrak{I}_{CL} \models_C \Psi_{\mathcal{J}}(\mathcal{G})$. (def. 4.2.5)

Case $\mathcal{G} = \mathcal{G}_1 \forall U \mathcal{G}_2$

The translation rule is

$\Psi_{\mathcal{J}}(\mathcal{G}_1 \forall U \mathcal{G}_2) = \forall p{:}\text{'W}\rightarrow\text{P'}\ \exists x{:}\text{'P}\rightarrow^{rt}\text{W'}\ (\Psi_{\mathcal{J}}(\mathcal{G}_2) \land \forall y{:}\text{'P}\rightarrow^{rt}\text{W'-x}\ (<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{J}}(\mathcal{G}_1)))$.

Let $p \in \text{'W}\rightarrow\text{P'}_{\mathcal{C}}$. There is an $x \in \text{'P}\rightarrow^{rt}\text{W'}_{\mathcal{C}}$ with $\mathfrak{I}[x(p(w))_{|W}] \models_M \mathcal{G}_2$ (★) and for every $y \in \text{'P}\rightarrow^{rt}\text{W'}_{\mathcal{C}}$ with $<_{\mathcal{C}}((p\circ y(w)),(p\circ x(w)))$: $\mathfrak{I}[y(p(w))_{|W}]_{\mathcal{C}} \models_M \mathcal{G}_1$ (✱).

Applying the induction hypothesis to (★) and exploiting $p \notin \Psi_{\mathcal{J}}(\mathcal{G}_2)$ and $x \notin \Psi_{\mathcal{J}}(\mathcal{G}_2)$, we get

$\mathfrak{I}_{CL}' := \mathfrak{I}_{CL}[p/p, x/x]_{\mathcal{V}}[WP/x(p(w))]_{\mathcal{C}}[p/w, x/p(w)]_{\mathcal{P}} \models_C \Psi_{\mathcal{J}}(\mathcal{G}_2)$. (★★)

Applying the induction hypothesis to (✱) and exploiting $p \notin \Psi_{\mathcal{J}}(\mathcal{G}_1)$ and $y \notin \Psi_{\mathcal{J}}(\mathcal{G}_1)$, we get for every $y \in \text{'P}\rightarrow^{rt}\text{W'}_{\mathcal{C}}$ with $<_{\mathcal{C}}((p\circ y)(w), (p\circ x)(w))$, $\mathfrak{I}_{CL}[p/p, y/y]_{\mathcal{V}}[WP/y(p(w))]_{\mathcal{C}}[p/w, y/p(w)]_{\mathcal{P}} \models_C \Psi_{\mathcal{J}}(\mathcal{G}_1)$, i.e. for every $y \in \text{'P}\rightarrow^{rt}\text{W'}_{\mathcal{C}}$

$$\mathfrak{I}_{CL}[p/p, x/x, y/y]_{\mathcal{V}}[WPy'(p(w))]_{\mathcal{C}}[p/w, x/p(w), y/p(w)]_{\mathcal{P}} \models_C <(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{J}}(\mathcal{G}_1).$$

Thus,. $\mathfrak{I}_{CL}' = \mathfrak{I}_{CL}[p/p,x/x]_{\mathcal{V}}[WP/x(p(w))]_{\mathcal{C}}[p/w,x/p(w)]_{\mathcal{P}} \models_C \forall y{:}\text{'P}\rightarrow^{rt}\text{W'-x}\ (<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{J}}(\mathcal{G}_1))$.

and with (★★): $\mathfrak{I}_{CL}' \models_C (\Psi_{\mathcal{J}}(\mathcal{G}_2) \land \forall y{:}\text{'P}\rightarrow^{rt}\text{W'-x}\ (<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{J}}(\mathcal{G}_1)))$

and from this

$\mathfrak{I}_{CL}[p/p]_{\mathcal{V}}[WP/p(w)]_{\mathcal{C}}[p/w]_{\mathcal{P}} \models_C \exists x{:}\text{'P}\rightarrow^{rt}\text{W'}\ (\Psi_{\mathcal{J}}(\mathcal{G}_2) \land \forall y{:}\text{'P}\rightarrow^{rt}\text{W'-x}\ (<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{J}}(\mathcal{G}_1)))$

and finally $\mathfrak{I}_{CL} \models_C \Psi_{\mathcal{J}}(\mathcal{G})$.

The case with the $\forall U^r$ operator is analogous to the previous one.

The cases with the dual modal operators are proved "dually" to the above cases.

The cases with the predicate logic connectives and quantifiers are trivial. ■

Theorem 5.2.13 Soundness of the Translation into OSPL

Every satisfiable MM-specification is translated into a satisfiable OSPL-specification.

Proof: The lemmas 5.2.10, 5.2.11 and 5.2.12 confirm that the translation from MM-Logic into CL is sound. The soundness or the translation from CL into OSPL is confirmed by the soundness lemma 4.3.10 for CL and the $\exists$-quantifier independency lemma 5.2.8 which guarantees the soundness of the strong Skolemization. ■

As mentioned earlier, we cannot achieve a complete calculus for full MM-Logic where $\mathfrak{R}^t$ is the transitive closure of $\mathfrak{R}^\emptyset$ because this is no first-order property. However, we can prove a weaker completeness result. It shows that the translation into CL is complete, i.e. the calculus with translation and refutation is complete, for all theorems which hold in nonstandard MM-Logic models where $\mathfrak{R}^t$ is more than the transitive closure of $\mathfrak{R}^\emptyset$. This means in particular that the calculus is complete as long as operators related to the $\mathfrak{R}^\emptyset$ and $\mathfrak{R}^r$ on one side and $\mathfrak{R}^t$ and $\mathfrak{R}^{rt}$ on the other side do not occur simultaneously.

Theorem 5.2.14 Weak Completeness

Whenever a translated MM-specification $\Psi_{\mathcal{S}}(\mathcal{S})$ is satisfied by a CL-interpretation $\mathfrak{I}_{CL} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, (<\mathcal{W}, \mathcal{P}>), \mathcal{P})$ then the original specification $\mathcal{S}$ is satisfied by the nonstandard MM-interpretation $\mathfrak{I} := \Psi_{\mathfrak{I}}^{-1}(\mathfrak{I}_{CL}) = ((W_{\mathcal{C}}, \mathfrak{R}), \mathcal{V}, \mathcal{W})$ where the transitive accessibility relation at least contains the transitive closure of the basic one.

Proof: We have to show:

1) $\mathfrak{I}$ is really an MM-interpretation, i.e. $\Psi_{\mathfrak{I}}^{-1}$ is well defined and

2) The MM-formulae in $\mathcal{S}$ are satisfied by $\mathfrak{I}$.

1) $\Psi_{\mathfrak{I}}^{-1}$ is well defined

The definition of $\Psi_{\mathfrak{I}}^{-1}$ (def. 5.2.4) is

"The **inverse interpretation morphism** $\Psi_{\mathfrak{I}}^{-1}$ generates from the CL-interpretation $\mathfrak{I}_{CL} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, (w_0), \mathcal{P})$ the MM-interpretation $\mathfrak{I} = ((W_{\mathcal{C}}, \mathfrak{R}), \mathcal{V}, PW_{\mathcal{C}}(w_0))$ where

a) $\mathfrak{R}^{\mathcal{R}} := \{(PW_{\mathcal{C}}(w), PW_{\mathcal{C}}(x(w))) \mid x \in \text{'P}\rightarrow^{\mathcal{R}}\text{W'}_{\mathcal{C}}\ w \in WP_{\mathcal{C}}\}$

For $w \in WP_{\mathcal{C}}$ let $\mathcal{P}(w) := \{PW_{\mathcal{C}}(x(w)) \mid x \in \text{'P}\rightarrow^{rt}\text{W'}_{\mathcal{C}}\}$. $\mathcal{P}(w)$ is the path described by w.

b) $\mathcal{P}(\mathcal{W}) := \{\mathcal{P}(w) \mid PW_{\mathcal{C}}(w) = \mathcal{W}\}$

c) A transition $\mathfrak{R}^{\mathcal{R}}(\mathcal{W}_1, \mathcal{W}_2)$ is labeled with a label ℓ iff $\exists\ x \in \text{'D,W}\rightarrow^{\mathcal{R}}\text{W'}_{\mathcal{C}}\ \exists w \in WP_{\mathcal{C}}$ with $\mathcal{W}_1 = PW_{\mathcal{C}}(w)$ and $\mathcal{W}_2 = PW_{\mathcal{C}}(x(\ell, w))$"

We have to check whether the axioms in def. 5.2.3 and the sort declarations in def. 5.2.1 are strong enough such that all conditions for M-frames (def. 5.1.3) are fulfilled (we can also use the derived formulae after lemma 5.2.4).

a) $\mathfrak{R}^{\emptyset}$ is discrete:

D5, F1, F3, and F4 are essentially the Péano axioms for natural numbers. Therefore paths are isomorphic to nonstandard models of natural numbers, i.e. in particular $\mathfrak{R}^{\emptyset}$ is discrete.

b) $\mathfrak{R}^r$ is the reflexive closure of $\mathfrak{R}^{\emptyset}$.

$\mathfrak{R}^r$ is reflexive because 'ID' $\sqsubseteq$ 'P$\rightarrow^r$W' implies that the identity function is in 'P$\rightarrow^{\emptyset}$W'$_{\mathcal{C}}$.

The derived clause F6 guarantees that $\mathfrak{R}^r$ contains nothing more than the reflexive closure.

c) $\mathfrak{R}^t$ is a transitive accessibility relation that includes the transitive closure of $\mathfrak{R}^{\emptyset}$.

To prove the transitivity, assume $\mathfrak{R}^t(\mathcal{W}_1, \mathcal{W}_2)$ and $\mathfrak{R}^t(\mathcal{W}_2, \mathcal{W}_3)$.

$\Rightarrow$ There are $w_1, w_2 \in WP_{\mathcal{C}}\ x, y \in$ 'P$\rightarrow^t$W'$_{\mathcal{C}}$ with $w_{1|W} = \mathcal{W}_1$, $x(w_1)_{|W} = \mathcal{W}_2$, $w_{2|W} = \mathcal{W}_2$ and $y(w_2)_{|W} = \mathcal{W}_3$,

$\Rightarrow\ w_2 = (x \circ PA_{\mathcal{C}}(w_2))(w_1)$	(C3)
Let $w' := MP_{\mathcal{C}}(x \circ PA_{\mathcal{C}}(w_2))(w_1)$	
$\Rightarrow\ w'_{|W} = \mathcal{W}_1$	(K1)
and $(MP_{\mathcal{C}}(x \circ PA_{\mathcal{C}}(w_2)) \circ x)(w_1)_{|W} = \mathcal{W}_2$	(D6)
$w_2 = MP_{\mathcal{C}}(x \circ PA_{\mathcal{C}}(w_2)) \circ MW_{\mathcal{C}}(x \circ PA(w_2))\ (w_1)$	(E1)
$= MW_{\mathcal{C}}(x \circ PA_{\mathcal{C}}(w_2))\ (w')$	(A3)
$= MW_{\mathcal{C}}(x) \circ MW_{\mathcal{C}}(PA(w_2))\ (w')$	(E1)
$= x(w')$	(D1, D2, B4)

Thus, we have $w'_{|W} = \mathcal{W}_1$, $x(w')_{|W} = w_{2|W} = \mathcal{W}_2$ and $y(w_2)_{|W} = y(x(w'))_{|W} = \mathcal{W}_3$ and because of the sort declaration $\circ$: 'P$\rightarrow^t$W'$\times$ 'P$\rightarrow^t$W' $\rightarrow$ 'P$\rightarrow^t$W', $x \circ y \in$ 'P$\rightarrow^t$W'$_{\mathcal{C}}$, and we can conclude $\mathfrak{R}^t(\mathcal{W}_1, \mathcal{W}_3)$.

To show that the transitive closure of $\mathfrak{R}^{\emptyset}$ is contained in $\mathfrak{R}^t$,

let $\mathfrak{R}^{\emptyset}(\mathcal{W}_1, \mathcal{W}_2), \mathfrak{R}^{\emptyset}(\mathcal{W}_2, \mathcal{W}_3), \ldots, \mathfrak{R}^{\emptyset}(\mathcal{W}_{n-1}, \mathcal{W}_n)$ be a sequence of $\mathfrak{R}^{\emptyset}$-transitions.

If $n = 2$ we exploit 'P$\rightarrow^{\emptyset}$W' $\sqsubseteq$ 'P$\rightarrow^t$W' and obtain immediately $\mathfrak{R}^t(\mathcal{W}_1, \mathcal{W}_2)$. If $n > 2$ we use the same arguments as above and exploit $\circ$: 'P$\rightarrow^{\emptyset}$W'$\times$ 'P$\rightarrow^{\emptyset}$W' $\rightarrow$ 'P$\rightarrow^t$W', finding $\mathfrak{R}^t(\mathcal{W}_1, \mathcal{W}_3)$. Induction on n and exploitation of $\circ$: 'P$\rightarrow^t$W'$\times$ 'P$\rightarrow^{\emptyset}$W' $\rightarrow$ 'P$\rightarrow^t$W' yields $\mathfrak{R}^t(\mathcal{W}_1, \mathcal{W}_n)$.

Axiom F5 ensures that each transitive transition is either a $\mathfrak{R}^{\emptyset}$-transition or it consists of another transitive transition followed by a $\mathfrak{R}^{\emptyset}$-transition.

d) $\mathfrak{R}^{rt}$ is the reflexive closure of $\mathfrak{R}^{t}$.
This is guaranteed because the identity is in 'P→^{rt}W'$_{\mathcal{C}}$ and because of axiom F2.

Since the 'P→$^{\mathcal{R}}$W'-functions are total, the $\mathfrak{R}^{\mathcal{R}}$-relations are serial.

Now we show that paths are correctly axiomatized.
Let $\mathcal{P} \in \mathfrak{P}(\mathcal{W})$ be a path, i.e. $\mathcal{P} = \mathcal{P}(w) := \{PW_{\mathcal{C}}(x(w)) \mid x \in$ 'P→^{rt}W'$_{\mathcal{C}}\}$ for some $w \in WP_{\mathcal{C}}$.
We have to show:

i) $\mathfrak{R}^{rt}$ is a total ordering on $\mathcal{P}$.
Let $\mathcal{W}_1, \mathcal{W}_2 \in \mathcal{P}$, i.e. $PW_{\mathcal{C}}(x_1(w)) = \mathcal{W}_1$ and $PW_{\mathcal{C}}(x_2(w)) = \mathcal{W}_2$ for some $x_1, x_2 \in$ 'P→^{rt}W'$_{\mathcal{C}}$
$\Rightarrow x_1(w) \leq_{\mathcal{C}} x_2(w)$ or $x_2(w) \leq_{\mathcal{C}} x_1(w)$ (axiom G4)
$\Rightarrow (x_1 \circ \twoheadrightarrow_{\mathcal{C}}(x_1, x_2))(w) = x_2(w)$ or $(x_2 \circ \twoheadrightarrow_{\mathcal{C}}(x_2, x_1))(w) = x_1(w)$ (axiom I2)
$\Rightarrow \twoheadrightarrow_{\mathcal{C}}(x_1, x_2)(x_1(w)) = x_2(w)$ or $\twoheadrightarrow_{\mathcal{C}}(x_2, x_1)(x_2(w)) = x_1(w)$ (axiom A2)
$\Rightarrow \mathfrak{R}^{rt}(\mathcal{W}_1, \mathcal{W}_2)$ or $\mathfrak{R}^{rt}(\mathcal{W}_2, \mathcal{W}_1)$
Suppose $\mathfrak{R}^{rt}(\mathcal{W}_1, \mathcal{W}_2)$
$\Leftrightarrow \twoheadrightarrow_{\mathcal{C}}(x_1, x_2)(x_1(w)) = x_2(w)$
$\Leftrightarrow x_1(w) \leq_{\mathcal{C}} x_2(w)$ (axiom I1)
Using the axioms G1 - G4 for $\leq$ we can conclude that $\mathfrak{R}^{rt}$ is actually a total ordering on $\mathcal{P}$.

ii) $\mathcal{W}$ is the smallest element.
$\mathcal{W} \in \mathcal{P}$ because $\mathcal{W} = PW_{\mathcal{C}}(ID_{\mathcal{C}}(w))$ and $ID_{\mathcal{C}} \in$ 'P→^{rt}W'$_{\mathcal{C}}$
$\mathcal{W}$ is the smallest element because $\mathfrak{R}^{rt}(\mathcal{W}, PW_{\mathcal{C}}(x(w)))$ for all $x \in$ 'P→^{rt}W'$_{\mathcal{C}}$

iii) $\mathcal{P}$ is maximal, i.e. there are no gaps in $\mathcal{P}$.
Let $\mathcal{W}_1 \in \mathcal{P}$, i.e. $PW_{\mathcal{C}}(x(w))$ for some $x \in$ 'P→^{rt}W'$_{\mathcal{C}}$
Since $x \circ +1 \in$ 'P→^{rt}W'$_{\mathcal{C}}$ $PW_{\mathcal{C}}((x \circ +1)(w)) \in \mathcal{P}$ as well.
Therefore each world in $\mathcal{P}$ has exactly one $\mathfrak{R}^{\emptyset}$-successor world in $\mathcal{P}$.

e) The seriality of labeled transitions is guaranteed by the totality of the 'D,W→$^{\mathcal{R}}$W'-functions.
Axiom J3 states that only the last label matters for labeled transitive transitions.

f) Because of axiom B2, the reflexive transition is labeled with all labels.

g) The fairness of labeled transitions on paths is ensured by the totality of the $BF_{\mathcal{C}}$ function which produces for each label ℓ a 'P→^{rt}W'-function that jumps to a world on the current path where $+1\mathcal{L}$ induces an ℓ-labeled transition that remains on the path. (axiom J2)

This finishes the necessary checks to confirm that $\mathfrak{I}$ is really an MM-interpretation.

2) The second part is to prove that the MM-formulae in $\mathcal{S}$ are satisfied by $\mathfrak{I} = \Psi_{\mathfrak{I}}^{-1}(\mathfrak{I}_{CL}) = ((W_{\mathcal{C}}\, \mathfrak{R}), \mathcal{V}, \mathcal{W}_0)$ where $\mathfrak{I}_{CL} = ((\mathcal{C}, \mathcal{SV}), \mathcal{V}, (w_0), \mathcal{P}), \mathcal{W}_0 = PW_{\mathcal{C}}(w_0) =: w_{0|W}$.
Therefore assume $\mathfrak{I}_{CL} \models_C \Psi_{\mathcal{F}}(\mathcal{G})$. We prove $\mathfrak{I} \models_M \mathcal{G}$ by induction on the structure of $\mathcal{G}$.
Base Case: $\mathcal{G}$ is an atom.
Since atoms are not modified by $\Psi_{\mathcal{F}}$ and since $\mathcal{W}_0$ is the actual Σ-structure for $\mathcal{G}$, $\mathfrak{I} \models_M \mathcal{G}$.
Induction Step: $\mathcal{G}$ is a non atomic formula.
The induction hypothesis is: For all subformulae $\mathcal{F}$ in $\Psi_{\mathcal{F}}(\mathcal{G})$, if a CL-interpretation $\mathfrak{I}'_{CL}$ satisfies $\mathcal{F}$ then the corresponding back translated MM-interpretation $\mathfrak{I}'$ satisfies $\mathcal{F}$.
We must perform a case analysis according to the top operator of $\mathcal{G}$.
Case $\mathcal{G} = \Box^{\mathcal{R}}\mathcal{F}$
The translation rule is $\Psi_{\mathcal{F}}(\Box^{\mathcal{R}}\mathcal{F}) = \forall x$:'W→$^{\mathcal{R}}$W' $\Psi_{\mathcal{F}}(\mathcal{F})$
Let $\mathfrak{R}^{\mathcal{R}}(\mathcal{W}_0, \mathcal{W}_1)$, $\mathcal{W}_1 \in W_{\mathcal{C}}$
$\Rightarrow \exists y \in$ 'P→$^{\mathcal{R}}$W'$_{\mathcal{C}}$, $w \in WP_{\mathcal{C}}$ with $PW_{\mathcal{C}}(w)_{|W} = \mathcal{W}_0$ and $PW_{\mathcal{C}}(y(w))_{|W} = \mathcal{W}_1$. (def. of $\mathfrak{R}^{\mathcal{R}}$)
$\Rightarrow w = PA_{\mathcal{C}}(w)(w_0)$. (axiom C3)

Let $\chi := (\mathrm{PA}_{\mathcal{C}}(w) \circ y) \in$ 'W$\rightarrow^{\mathcal{R}}W'_{\mathcal{C}}$

$\Rightarrow$ $\chi(w_0)_{|W} = (\mathrm{PA}_{\mathcal{C}}(w) \circ y)(w_0)_{|W} = y(w)_{|W} = \mathcal{W}_1$

and $\Im_{CL}[x/\chi]_{\mathcal{V}}[WP/\chi(w_0)]_{\mathcal{C}}[x/w_0]_{\mathcal{P}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ (def.4.2.5, $\Im_{CL} \models_C \forall x$:'W$\rightarrow^{\mathcal{R}}$W' $\Psi_{\mathcal{F}}(\mathcal{F})$)

$\Rightarrow$ $\Im[\mathcal{W}_1] \models_M \mathcal{F}$ (induction hypothesis)

$\Rightarrow$ $\Im \models_M \mathcal{G}$ (def. 5.1.5)

Case $\mathcal{G} = [\![t]\!]^{\mathcal{R}}\mathcal{F}$

The translation rule is $\Psi_{\mathcal{F}}([\![t]\!]^{\mathcal{R}}\mathcal{F}) = \forall\downarrow$(x:'D,W$\rightarrow^{\mathcal{R}}$W', z:S(t)=t) $\Psi_{\mathcal{F}}(\mathcal{F})$.

Since terms are not modified by $\Psi_{\mathcal{F}}$ and since $\mathcal{W}_0$ is the actual Σ-structure for t, $\Im(t) = \Im_{CL}(t)$.

Let $(\mathcal{W}_0, \mathcal{W}_1) \in \Re^{\mathcal{R}}(\Im(t))$, $\mathcal{W}_1 \in W_{\mathcal{C}}$

$\Rightarrow$ There is an $\chi \in$ 'D,W$\rightarrow^{\mathcal{R}}W'_{\mathcal{C}}$ with $\chi(\Im(t))(w_0)_{|W} = \mathcal{W}_1$

$\Rightarrow$ $\Im_{CL}[x/\chi]_{\mathcal{V}}[WP/\chi(\Im(t), w_0)]_{\mathcal{C}}[x/w_0]_{\mathcal{P}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ (def.4.2.5)

$\Rightarrow$ $\Im[\mathcal{W}_1] \models_M \mathcal{F}$ (induction hypothesis)

$\Rightarrow$ $\Im \models_M \mathcal{G}$ (def. 5.1.5)

Case $\mathcal{G} = \blacktriangleright\mathcal{F}$

The translation rule is $\Psi_{\mathcal{F}}(\blacktriangleright\mathcal{F}) = \forall p$:'W$\rightarrow$P' $\exists x$:'P$\rightarrow^{rt}$W' $\Psi_{\mathcal{F}}(\mathcal{F})$

Let $\mathcal{P} \in \mathfrak{P}(\mathcal{W}_0)$ be a path.

$\Rightarrow$ There is a $w \in \mathrm{WP}_{\mathcal{C}}$ with $w_{|W} = \mathcal{W}_0$ and $\mathcal{P} = \{\chi(w)_{|W} \mid \chi \in$ 'P$\rightarrow^{rt}$W'$_{\mathcal{C}}\}$

$\Rightarrow$ $w = \mathrm{PA}_{\mathcal{C}}(w)(w_0)$ (axiom C3)

$\Rightarrow$ $\Im_{CL}[p/\mathrm{PA}_{\mathcal{C}}(w)]_{\mathcal{V}}[WP/w]_{\mathcal{C}}[p/w_0]_{\mathcal{P}} \models_C \exists x$:'P$\rightarrow^{rt}$W' $\Psi_{\mathcal{F}}(\mathcal{F})$ (def.4.2.5)

$\Rightarrow$ $\Im_{CL}[p/\mathrm{PA}_{\mathcal{C}}(w), x/\chi]_{\mathcal{V}}[WP/\chi(w)]_{\mathcal{C}}[p/w_0, x/w]_{\mathcal{P}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ for some $\chi \in$ 'P$\rightarrow^{rt}$W'$_{\mathcal{C}}$ (def.4.2.5)

$\Rightarrow$ $\Im[\chi(w)_{|W}] \models_M \mathcal{F}$ and $\chi(w)_{|W} \in \mathcal{P}$ (induction hypothesis)

$\Rightarrow$ $\Im \models_M \mathcal{G}$ (def. 5.1.5)

Case $\mathcal{G} = |t)\mathcal{F}$

The translation rule is $\Psi_{\mathcal{F}}(|t)\mathcal{F}) = \forall p$:'W$\rightarrow$P' $\wp(\mathrm{BF}(t) \circ +1)\, \Psi_{\mathcal{F}}(\mathcal{F})$

Let $\mathcal{P} \in \mathfrak{P}(\mathcal{W}_0)$ be a path.

$\Rightarrow$ There is a $w \in \mathrm{WP}_{\mathcal{C}}$ with $w_{|W} = \mathcal{W}_0$ and $\mathcal{P} = \{\chi(w)_{|W} \mid \chi \in$ 'P$\rightarrow^{rt}$W'$_{\mathcal{C}}\}$

$\Rightarrow$ $w = \mathrm{PA}_{\mathcal{C}}(w)(w_0)$ (axiom C3)

$\Rightarrow$ $\Im'_{CL} := \Im_{CL}[p/\mathrm{PA}_{\mathcal{C}}(w)]_{\mathcal{V}}[WP/w]_{\mathcal{C}}[p/w_0]_{\mathcal{P}} \models_C \wp(\mathrm{BF}(t) \circ +1)\ \Psi_{\mathcal{F}}(\mathcal{F})$ (def.4.2.5)

Let $w_1 := \Im'_{CL}(\mathrm{BF}(t) \circ +1))(w)$

$\Rightarrow$ $\Im_{CL}[p/\mathrm{PA}_{\mathcal{C}}(w)]_{\mathcal{V}}[WP/w_1]_{\mathcal{C}}[p/w_0]_{\mathcal{P}} \models_C \Psi_{\mathcal{F}}(\mathcal{F})$ (def.4.2.5)

$\Rightarrow$ $\Im[w_{1|W}] \models_M \mathcal{F}$ (induction hypothesis)

and $w_{1|W}$ is the first world on $\mathcal{P}$ after a transition labeled with $\Im'_{CL}(t) = \Im(t)$ (axiom J1)

$\Rightarrow$ $\Im \models_M \mathcal{G}$ (def. 5.1.5)

Case $\mathcal{G} = \mathcal{F}_1 \forall U\, \mathcal{F}_2$

The translation rule is

$\Psi_{\mathcal{F}}(\mathcal{F}_1 \forall U\, \mathcal{F}_2) = \forall p$:'W$\rightarrow$P' $\exists x$:'P$\rightarrow^{rt}$W' $(\Psi_{\mathcal{F}}(\mathcal{F}_2) \wedge \forall y$:'P$\rightarrow^{rt}$W'-x $(<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}_1))$.

Let $\mathcal{P} \in \mathfrak{P}(\mathcal{W}_0)$ be a path.

$\Rightarrow$ There is a $w \in \mathrm{WP}_{\mathcal{C}}$ with $w_{|W} = \mathcal{W}_0$ and $\mathcal{P} = \{\chi(w)_{|W} \mid \chi \in$ 'P$\rightarrow^{rt}$W'$_{\mathcal{C}}\}$

$\Rightarrow$ $w = \mathrm{PA}_{\mathcal{C}}(w)(w_0)$ (axiom C3)

$\Rightarrow$ $\Im'_{CL} := \Im_{CL}[p/\mathrm{PA}_{\mathcal{C}}(w)]_{\mathcal{V}}[WP/w]_{\mathcal{C}}[p/w_0]_{\mathcal{P}}$

$\models_C \exists x$:'P$\rightarrow^{rt}$W' $(\Psi_{\mathcal{F}}(\mathcal{G}) \wedge \forall y$:'P$\rightarrow^{rt}$W'-x $(<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}))$. (def.4.2.5)

$\Rightarrow$ $\Im''_{CL} := \Im_{CL}[p/\mathrm{PA}_{\mathcal{C}}(w), x/\chi]_{\mathcal{V}}[WP/\chi(w)]_{\mathcal{C}}[p/w_0, x/w]_{\mathcal{P}}$ (def.4.2.5)

$\models_C (\Psi_{\mathcal{F}}(\mathcal{F}_2) \wedge \forall y$:'P$\rightarrow^{rt}$W'-x $(<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}_1))$ for some $\chi \in$ 'P$\rightarrow^{rt}$W'$_{\mathcal{C}}$

$\Rightarrow$ $\Im''_{CL} \models_C \Psi_{\mathcal{F}}(\mathcal{F}_2)$ and $\Im''_{CL} \models_C \forall y$:'P$\rightarrow^{rt}$W'-x $(<(p\circ y, p\circ x) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}_1))$ (✣) (def.4.2.5)

$\Rightarrow$ $\Im[\chi(w)_{|W}] \models_M \mathcal{F}_2$ and $\mathcal{W}_2 := \chi(w)_{|W} \in \mathcal{P}$ (✱) (induction hypothesis)

Let $\mathcal{W}_1 \in \mathcal{P}$, $\mathcal{W}_1 \neq \mathcal{W}_2$ and $\Re^t(\mathcal{W}_1,\mathcal{W}_2)$

$\Rightarrow$ $\exists w_1 \in \mathrm{WP}_{\mathcal{C}}$ with $w_{1|\mathrm{W}} = \mathcal{W}_1$ and $z(w_1) = x(w)$ for some $z \in$ '$\mathrm{P}\rightarrow^{\mathrm{rt}}\mathrm{W}$'$_{\mathcal{C}}$ (def. of $\Re^t$)

and $w_1 = y(w)$ for some $y \in$ '$\mathrm{P}\rightarrow^{\mathrm{rt}}\mathrm{W}$'$_{\mathcal{C}}$ ($w_1 \in \mathcal{P}$)

$\Rightarrow$ $z(y(w)) = x(w)$

$\Rightarrow$ $y(w) \leq_{\mathcal{C}} x(w)$ (axiom I1)

$\Rightarrow$ $y(w) <_{\mathcal{C}} x(w)$ (★) ($\mathcal{W}_2 = x(w)_{|\mathrm{W}} \neq y(w)_{|\mathrm{W}} = \mathcal{W}_2$ and axiom H3)

$\mathfrak{I}''{}'_{\mathrm{CL}} := \mathfrak{I}''_{\mathrm{CL}}[\mathrm{y}/y]_{\mathcal{V}}[\mathrm{WP}/y(\mathcal{P}(\mathrm{x}))]_{\mathcal{C}}[\mathrm{y}/\mathcal{P}(\mathrm{x})]_{\mathcal{P}} \models_{\mathrm{C}} <(\mathrm{p}\circ\mathrm{y}, \mathrm{p}\circ\mathrm{x}) \Rightarrow \Psi_{\mathcal{F}}(\mathcal{F}_1)$ (✣ and def.4.2.5)

$\Rightarrow$ $\mathfrak{I}_{\mathrm{CL}}[\mathrm{p}/\mathrm{PA}_{\mathcal{C}}(w),\mathrm{x}/x, \mathrm{y}/y]_{\mathcal{V}}[\mathrm{WP}/y(w)]_{\mathcal{C}}[\mathrm{p}/w_0,\mathrm{x}/w,\mathrm{y}/w]_{\mathcal{P}} \models_{\mathrm{C}} <(\mathrm{p}\circ\mathrm{y}, \mathrm{p}\circ\mathrm{x})\Rightarrow\Psi_{\mathcal{F}}(\mathcal{F}_1)$ (def. of $\mathfrak{I}''$)

$\mathfrak{I}''{}'_{\mathrm{CL}} \models_{\mathrm{C}} <(\mathrm{p}\circ\mathrm{y}, \mathrm{p}\circ\mathrm{x})$ (★ and def. 4.2.5)

$\Rightarrow$ $\mathfrak{I}''{}'_{\mathrm{CL}} \models_{\mathrm{C}} \Psi_{\mathcal{F}}(\mathcal{F}_1)$

$\Rightarrow$ $\mathfrak{I}[x(w)_{|\mathrm{W}}] = \mathfrak{I}[\mathcal{W}_2] \models_{\mathrm{M}} \mathcal{F}_2$

$\Rightarrow$ $\mathfrak{I} \models_{\mathrm{M}} \mathcal{G}$ (✶ and def. 5.1.5)

The proofs for the other cases are similar. ■

This completeness result together with the completeness of the translation from Context Logic into OSPL (lemma 4.3.15) means that theorem proving in MM-Logic by translation via CL into OSPL is (weakly) complete.

5.3 Examples

The following examples for theorem proving with translation via CL into OSPL and refutation with resolution and paramodulation in OSPL are chosen to illustrate typical applications of clauses axiomatizing the possible worlds structure of MM-Logic (def. 5.2.3). We use first-order syntax now, but except in example 5.3.5, we drop the PW function which, according to the formula morphism $\Pi_{\mathcal{F}}$, embraces all the WP-terms in the translated domain terms and atoms. Keeping this function does not change the deductions in our examples. Furthermore we assume the associativity of ∘ and the axiom A3: (x∘y)(w) = y(x(w)) to be built into the unification algorithm.

Notice that the variables in different clauses are always different, although we usually choose common names.

Example 5.3.1

The first example proves that in the modal system D (no special properties of the accessibility relation except seriality) Löb's Axioms $\Box(\Box\mathcal{G} \Rightarrow \mathcal{G}) \Rightarrow \Box\mathcal{G}$ imply the formula $\Box Q \Rightarrow \Box\Box Q$ that characterizes transitive accessibility relations. Actually this holds also in K (nonserial accessibility relation) and there Löb's Axioms axiomatize the modal system G that has a transitive and non-serial accessibility relation $\mathfrak{R}$ with no infinite $\mathfrak{R}$-chains. Let $\mathcal{G} := Q \wedge \Box Q$. The theorem to be proved is

$$\mathcal{F} := (\Box^{\emptyset}(\Box^{\emptyset}(Q \wedge \Box^{\emptyset}Q) \Rightarrow (Q \wedge \Box^{\emptyset}Q)) \Rightarrow \Box^{\emptyset}(Q \wedge \Box^{\emptyset}Q)) \Rightarrow (\Box^{\emptyset}Q \Rightarrow \Box^{\emptyset}\Box^{\emptyset}Q)$$

($\Box^{\emptyset}$ is the MM-Logic operator that corresponds to the system D modal logic operator $\Box$)

Translation of the negated formula into CL (def. 5.2.2) yields

$$\neg(\forall u\, (\forall a\, (Q \wedge \forall v\, Q) \Rightarrow (Q \wedge \forall bQ)) \Rightarrow \forall w(Q \wedge \forall xQ)) \Rightarrow (\forall yQ \Rightarrow \forall c,dQ)$$

All variables are of type '$W \rightarrow^{\emptyset} W$'.

The negation normal form is

$$(\forall u\, (\exists a\, (Q \wedge \forall v\, Q) \wedge (\neg Q \vee \exists b \neg Q)) \vee \forall w(Q \wedge \forall xQ)) \wedge (\forall yQ \wedge \exists c,d \neg Q)$$

Translation into OSPL (def. 4.3.2) yields

$$((\forall u\, (Q[au] \wedge \forall v Q[auv]) \wedge (\neg Q[a] \vee \neg Q[ab])) \vee \forall w(Q[w] \wedge \forall x\, Q[wx])) \wedge (\forall y\, Q[y] \wedge \neg Q[cd])$$

To make it more readable we abbreviate $PW(\downarrow(x_1 \circ \ldots \circ x_n), 0)$ with $[x_1 \ldots x_n]$.

The clauses are:

R1: Q[au], P[w]	R2: Q[au], Q[wx]	R3: Q[auv], Q[w]
R4: Q[auv], P[wx]	R5: ¬Q[a], ¬Q[ab], Q[w]	R6: ¬Q[a], ¬Q[ab], Q[wx]
R7: Q[y]	R8: ¬Q[cd].	

a,b,c,d are constants of type '$W \rightarrow^{\emptyset} W$', x,y,z,u,v,w are variables of type '$W \rightarrow^{\emptyset} W$'.

Resolution: R6,1 & R7, $\sigma = \{y \mapsto a\}$

→ R9: ¬Q[ab], Q[wx]

Resolution: R9,2 & R8, $\sigma = \{w \mapsto c, x \mapsto d\}$

→ R10: ¬Q[ab]

Resolution: R10 & R2,1, $\sigma = \{u \mapsto b\}$

→ R11: Q[wx].

Resolution: R11 & R8, $\sigma = \{w \mapsto c, x \mapsto d\}$

→ R12: empty clause. ■

None of the axioms for the possible worlds structure were necessary. This is always the case when $\Box^{\emptyset}$ and $\Diamond^{\emptyset}$ are the only modal operators occurring in the formulae. In case $\Box^{r}$ and $\Diamond^{r}$ are the only operators occurring (modal logic T or M), ID ∘ x = x and x ∘ ID = x are the only axioms needed. In case $\Box^{t}$ and $\Diamond^{t}$ are the only operators occurring (modal logic D4), the associativity of ∘ is needed and finally in case $\Box^{rt}$ and $\Diamond^{rt}$ are the only operators occurring, ID ∘ x = x and x ∘ ID = x together with the associativity of ∘ are needed.

Example 5.3.2 **Axiom F2**

We want to prove $Q \wedge \Box^{\emptyset}Q \Rightarrow \Box^{r}Q$.

Translation of the negated formula into CL (def. 5.2.2) yields

$$\neg(Q \wedge (\forall x{:}\text{'}W \to^{\emptyset} W\text{'}\ Q) \Rightarrow \forall y{:}\text{'}W \to^{r} W\text{'}\ Q)$$

The negation normal form is:

$$Q \wedge (\forall x{:}\text{'}W \to^{\emptyset} W\text{'}\ Q) \wedge \exists y{:}\text{'}W \to^{r} W\text{'}\ \neg Q$$

Translation into OSPL (def. 4.3.2) yields

$$Q(0) \wedge (\forall x{:}\text{'}W \to^{\emptyset} W\text{'}\ Q{\downarrow}(x, 0) \wedge \neg Q{\downarrow}(a, 0) \qquad a{:}\text{'}W \to^{r} W\text{'}$$

(We simplified PW(↓(ID, 0)) to 0)

The clauses are:

F2 $\forall x{:}\text{'}W \to^{rt} W\text{'}\ \forall w{:}WP\ \ {\downarrow}(x, w) = w \vee {\downarrow}(x, w) = {\downarrow}(\text{-}R(x), w)$.

R1 $Q(0)$ R2 $\forall x{:}\text{'}W \to^{\emptyset} W\text{'}\ Q{\downarrow}(x, 0)$ R3 $\neg Q{\downarrow}(a, 0)$

Resolution between R2 and R3 is not possible because the sort of x is weaker than the sort of a.

Paramodulation: F2,1,r & R1, $\sigma = \{w \mapsto 0\}$

→ R4: $\forall x{:}\text{'}W \to^{rt} W\text{'}\ Q{\downarrow}(x, 0) \vee {\downarrow}(x, 0) = {\downarrow}(\text{-}R(x), 0)$.

Resolution: R3 & R4,1, $\sigma = \{x \mapsto a\}$

→ R5: ${\downarrow}(a, 0) = {\downarrow}(\text{-}R(a), 0)$.

Paramodulation: R5 & R3, $\sigma = \emptyset$

→ R6: $\neg Q{\downarrow}(\text{-}R(a), 0)$

Resolution: R6 & R2, $\sigma = \{x \mapsto a\}$, The sort of '-R(a)' is '$W \to^{\emptyset} W$'

→ R7: empty clause.

In the same way we can prove $P \wedge \Box^{t}P \Rightarrow \Box^{rt}P$. ■

The example has shown that the axiom F2 is necessary when statements correlate □-operators with and without reflexivity.

Example 5.3.3 **Axiom A4**

We want to prove $\forall x{:}D\ [\![x]\!]^{t}Q \Rightarrow [\![x]\!]^{t}[\![x]\!]^{t}Q$ (transitivity axiom in logic D4)

Translation of the negated formula into CL (def. 5.2.2) yields

$$\neg(\forall x{:}D\ (\forall(y{:}\text{'}D,W \to^{t} W\text{'}, u_1{=}x)\ Q) \Rightarrow \forall(a{:}\text{'}D,W \to^{t} W\text{'}, u_2{=}x)\ \forall(b{:}\text{'}D,W \to^{t} W\text{'}, u_3{=}x)\ Q)$$

The negation normal form is:

$$\forall x{:}D\ (\forall(y{:}\text{'}D,W \to^{t} W\text{'}, u_1{=}x)\ Q) \vee \exists(a{:}\text{'}D,W \to^{t} W\text{'}, u_2{=}x)\ \exists(b{:}\text{'}D,W \to^{t} W\text{'}, u_3{=}x)\ \neg Q$$

Translation into OSPL (def. 4.3.2) yields

$$\forall x{:}D\ \forall y{:}\text{'}D,W \to^{t} W\text{'}\ Q{\downarrow}({\downarrow}(y, x), 0) \wedge \neg Q{\downarrow}({\downarrow}(a, x) \circ {\downarrow}(b, x)), 0)$$

$a{:}\text{'}D,W \to^{t} W\text{'}$ and $b{:}\text{'}D,W \to^{t} W\text{'}$

The clauses are:

A4 $\forall x,y{:}\text{'}D,W \to^{t} W\text{'}\ \forall l{:}D\ \ {\downarrow}(x \circ y, l) = {\downarrow}(x, l) \circ {\downarrow}(y, l)$

R1 $\forall x{:}D\ \forall y{:}\text{'}D,W \to^{t} W\text{'}\ Q{\downarrow}({\downarrow}(y, x), 0)$ R2 $\forall x{:}D\ \neg Q{\downarrow}({\downarrow}(a, x) \circ {\downarrow}(b, x)), 0)$

Paramodulation A4,r & R2 $\sigma = \{x_{A4} \mapsto a, y \mapsto b, l \mapsto x_{R2}\}$

→ R3: $\forall x{:}D\ \neg Q{\downarrow}({\downarrow}(a \circ b, x), 0)$

The sort of $a \circ b$ is '$D,W \to^{t} W$'. Therefore R3 and R1 unify and we get the empty clause. ■

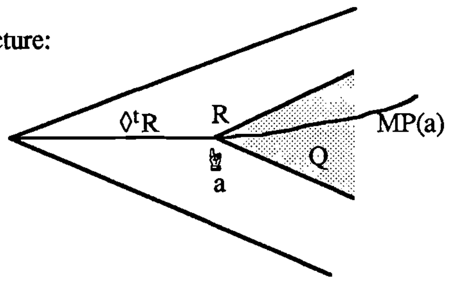
The next examples illustrate some correlations between the usual modal operators and those involving paths through possible worlds structures.

Example 5.3.4 Axioms G4, I2, E1

We want to prove $\Diamond^{t}(\Box^{rt}Q \wedge R) \Rightarrow \twoheadrightarrow(Q \vee \Diamond^{t}R)$

The interesting part of the possible worlds structure:
'a' corresponds to the $\Diamond$-operator and
MP(a) corresponds to the $\twoheadrightarrow$-operator

The path crossing 'a'
satisfies $\Diamond^{t}R$ (before a) or Q (after a)



Translation of the negated formula into CL (def. 5.2.2) yields

$$\neg((\exists a{:}'W\to^{t}W'\ (\forall x{:}'W\to^{rt}W'\ Q) \wedge R)) \Rightarrow \exists p{:}'W\to P'\ \forall b{:}'P\to^{rt}W'\ (Q \vee \exists y{:}'W\to^{t}W'\ R)$$

The negation normal form is:

$$(\exists a{:}'W\to^{t}W'\ (\forall x{:}'W\to^{rt}W'\ Q)) \wedge R \wedge \forall p{:}'W\to P'\ \exists b{:}'P\to^{rt}W'\ (\neg Q \wedge \forall y{:}'W\to^{t}W'\ \neg R)$$

Translation into OSPL (def. 4.3.2) yields

$$\forall x{:}'W\to^{rt}W'\ Q{\downarrow}(a{\circ}x, 0) \wedge R{\downarrow}(a, 0) \wedge \forall p{:}'W\to P'\neg Q{\downarrow}(p{\circ}b, 0)$$
$$\wedge\ \forall y{:}'W\to^{t}W'\neg R{\downarrow}(p{\circ}b{\circ}y, 0)$$
$$a{:}'W\to^{t}W',\ b{:}'P\to^{rt}W'$$

The clauses are:

G4 $\forall x,y{:}'P\to^{rt}W'\ \forall w{:}WP\quad {\downarrow}(x, w) \le {\downarrow}(y, w)\ \vee\ {\downarrow}(y, w) \le {\downarrow}(x, w).$

I2 $\forall x,y{:}'P\to^{rt}W'\ \forall w{:}WP\quad \neg\, {\downarrow}(x, w) \le {\downarrow}(y, w) \vee {\downarrow}(x \circ\rightarrowtail(x, y)), w) = {\downarrow}(y, w)$

E1 $\forall x{:}'W\to^{rt}W'\quad x = MP(x) \circ MW(x)$

R1 $\forall x{:}'W\to^{rt}W'\ Q{\downarrow}(a{\circ}x, 0)$

R2 $R{\downarrow}(a, 0)$

R3 $\forall p{:}'W\to P'\ \neg Q{\downarrow}(p{\circ}b, 0)$

R4 $\forall p{:}'W\to P'\ \forall y{:}'W\to^{t}W'\ \neg R{\downarrow}(p{\circ}b{\circ}y, 0)$

Paramodulation I2,2r & R3, $\sigma = \{y \mapsto b, w \mapsto {\downarrow}(p, 0))$

$\to$ R5: $\forall p{:}'W\to P'\ \forall x{:}'P\to^{rt}W'\neg\, {\downarrow}(p{\circ}x, 0) \le {\downarrow}(p{\circ}b, 0) \vee \neg Q{\downarrow}(p{\circ}x{\circ}\rightarrowtail(x, b), 0)$

Paramodulation E1,l & R1, $\sigma = \{x_{E1} \mapsto a\}$

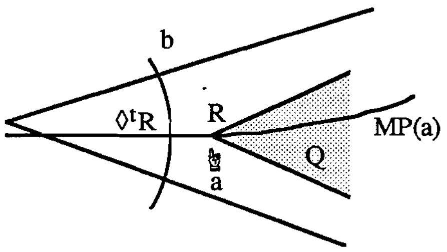
$\to$ R6: $\forall x{:}'W\to^{rt}W'\ Q{\downarrow}(MP(a){\circ}MW(a){\circ}x, 0)$

Resolution R5,2 & R6, $\sigma = \{p \mapsto MP(a), x_{R5} \mapsto MW(a), x_{R6} \mapsto \rightarrowtail(MW(a), b)\}$

$\to$ R7: $\neg\, {\downarrow}(MP(a){\circ}MW(a), 0) \le {\downarrow}(MP(a){\circ}b, 0)$

We have derived that b precedes a.
The next step will be to derive
the opposite.

b corresponds to the negated
$\twoheadrightarrow$-operator in the theorem



Paramodulation E1,l & R2, $\sigma = \{x \mapsto a\}$

$\to$ R8: $R{\downarrow}(MP(a){\circ}MW(a), 0)$

Paramodulation I2,2r & R8, $\sigma = \{y \mapsto MW(a), w \mapsto {\downarrow}(p, 0)\}$

$\to$ R9: $\forall x{:}'P\to^{rt}W'\neg\, {\downarrow}(MP(a){\circ}x, 0) \le {\downarrow}(MP(a){\circ}MW(a), 0) \vee R{\downarrow}(MP(a){\circ}x{\circ}\rightarrowtail(x, MW(a)), 0)$

Resolution R4 & R9,2, $\sigma = \{p \mapsto MP(a), x \mapsto b, y \mapsto \rightarrowtail(b, MW(a))\}$

$\to$ R10: $\neg\, {\downarrow}(MP(a){\circ}b, 0) \le {\downarrow}(MP(a){\circ}MW(a), 0)$

Two further resolutions with G4, R7 and R10 yield the empty clause. ■

Example 5.3.5 **Axioms B3, C2, D2, D13, E1, E4, G4, I2, K1**

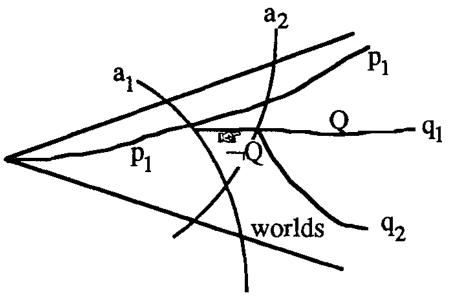
We want to prove $\blacktriangleright\!\twoheadrightarrow Q \Rightarrow \twoheadrightarrow\!\blacktriangleright Q$

The negation of this formula in MM-syntax is $\blacktriangleright\!\twoheadrightarrow Q \wedge \blacktriangleright\!\twoheadrightarrow\neg Q$.

Although the formula looks simple, the situation it describes in terms of the possible worlds structure is quite complex and therefore the arguments for the refutation are also are quite complex. As an orientation for following the refutation the interesting part of the possible worlds structure is drawn in the figure below.

a_1 corresponds to the first $\blacktriangleright$
a_2 corresponds to the second $\blacktriangleright$
p_1 corresponds to the first $\twoheadrightarrow$

The contradiction, i.e. Q and ¬Q in the part where a_1 is before a_2 occurs at the ramification of q_1 and q_2.



Translation of the negated formula into CL (def. 5.2.2) yields

$$\neg((\forall p_1{:}\text{'}W{\to}P\text{'}\ \exists a_1{:}\text{'}P{\to}^{rt}W\text{'}\ \exists q_1{:}\text{'}W{\to}P\text{'}\ \forall y_1{:}\text{'}P{\to}^{rt}W\text{'}\ Q)$$
$$\Rightarrow \exists p_2{:}\text{'}W{\to}P\text{'}\ \forall a_2{:}\text{'}P{\to}^{rt}W\text{'}\ \forall q_2{:}\text{'}W{\to}P\text{'}\ \exists y_2{:}\text{'}P{\to}^{rt}W\text{'}\ Q$$

The negation normal form is:

$$(\forall p_1{:}\text{'}W{\to}P\text{'}\ \exists a_1{:}\text{'}P{\to}^{rt}W\text{'}\ \exists q_1{:}\text{'}W{\to}P\text{'}\ \forall y_1{:}\text{'}P{\to}^{rt}W\text{'}\ Q)$$
$$\wedge\ \forall p_2{:}\text{'}W{\to}P\text{'}\ \exists a_2{:}\text{'}P{\to}^{rt}W\text{'}\ \exists q_2{:}\text{'}W{\to}P\text{'}\ \forall y_2{:}\text{'}P{\to}^{rt}W\text{'}\ \neg Q$$

This time we need the correct translation with the PW function.

Translation into OSPL (def. 4.3.2) yields

$$\forall p_1{:}\text{'}W{\to}P\text{'}\ \forall y_1{:}\text{'}P{\to}^{rt}W\text{'}\ Q(PW(\downarrow(p_1{\circ}a_1{\circ}q_1{\circ}y_1, 0)))$$
$$\wedge\ \forall p_2{:}\text{'}W{\to}P\text{'}\ \forall y_2{:}\text{'}P{\to}^{rt}W\text{'}\ \neg Q(PW(\downarrow(p_2{\circ}a_2{\circ}q_2{\circ}y_2, 0)))$$
$$a_1{:}\text{'}P{\to}^{rt}W\text{'},\ a_2{:}\text{'}P{\to}^{rt}W\text{'},\ q_1{:}\text{'}W{\to}P\text{'},\ q_2{:}\text{'}W{\to}P\text{'}$$

The clauses are:

B3	$\forall x{:}\text{'}W{\to}^{rt}W\text{'}$	$x \circ ID = x$
C2	$\forall p{:}\text{'}W{\to}P\text{'}$	$MW(p) = ID$
D2	$\forall x{:}\text{'}P{\to}^{rt}W$	$MW(x) = x$
D13	$\forall x,y,z{:}\text{'}P{\to}^{rt}W\text{'}\ \forall p{:}\text{'}W{\to}P\text{'}\ \forall w{:}WP$	$\neg \downarrow(x, w) \le \downarrow(y, w) \vee \downarrow(MP(x{\circ}z{\circ}p){\circ}x),w) \le \downarrow(MP(x{\circ}z{\circ}p){\circ}y),w)$
E1	$\forall x{:}\text{'}W{\to}^{rt}W\text{'}$	$x = MP(x) \circ MP(x)$
E4	$\forall x,y{:}\text{'}W{\to}^{rt}W\text{'}$	$MW(x{\circ}y) = MW(x) \circ MW(y)$
G4	$\forall x,y{:}\text{'}P{\to}^{rt}W\text{'}\ \forall w{:}WP$	$\downarrow(x, w) \le \downarrow(y, w) \ \vee\ \downarrow(y, w) \le \downarrow(x, w)$.
I2	$\forall x,y{:}\text{'}P{\to}^{rt}W\text{'}\ \forall w{:}WP$	$\neg \downarrow(x, w) \le \downarrow(y, w) \vee \downarrow(x \circ\!\!\twoheadrightarrow(x, y)), w) = \downarrow(y, w)$
K1	$\forall x{:}\text{'}W{\to}P\text{'}\ \forall w{:}WP$	$PW(\downarrow(x, w)) = PW(w)$

R1 $\forall p_1{:}\text{'}W{\to}P\text{'}\ \forall y_1\text{'}P{\to}^{rt}W\text{'}\ Q(PW(\downarrow(p_1{\circ}a_1{\circ}q_1{\circ}y_1, 0)))$

R2 $\forall p_2{:}\text{'}W{\to}P\text{'}\ \forall y_2\text{'}P{\to}^{rt}W\text{'}\ \neg Q(PW(\downarrow(p_2{\circ}a_2{\circ}q_2{\circ}y_2, 0)))$

Paramodulation E1,1 & R1, $\sigma = \{x \mapsto a_1{\circ}q_1\}$

$\to$ R3: $\forall p_1{:}\text{'}W{\to}P\text{'}\ \forall y_1\text{'}P{\to}^{rt}W\text{'}\ Q(PW(\downarrow(p_1{\circ}MP(a_1{\circ}q_1){\circ}MW(a_1{\circ}q_1){\circ}y_1, 0)))$

Paramodulation E4,1 & R3, $\sigma = \{x \mapsto a_1, y \mapsto q_1\}$

$\to$ R4: $\forall p_1{:}\text{'}W{\to}P\text{'}\ \forall y_1\text{'}P{\to}^{rt}W\text{'}\ Q(PW(\downarrow(p_1{\circ}MP(a_1{\circ}q_1){\circ}MW(a_1){\circ}MW(q_1){\circ}y_1, 0)))$

Paramodulation C2,1 & R4, $\sigma = \{p \mapsto q_1\}$

$\to$ R4: $\forall p_1{:}\text{'}W{\to}P\text{'}\ \forall y_1\text{'}P{\to}^{rt}W\text{'}\ Q(PW(\downarrow(p_1{\circ}MP(a_1{\circ}q_1){\circ}MW(a_1){\circ}ID{\circ}y_1, 0)))$

Paramodulation D2,1 & R4, $\sigma = \{x \mapsto a_1\}$

$\to$ R5: $\forall p_1{:}\text{'}W{\to}P\text{'}\ \forall y_1\text{'}P{\to}^{rt}W\text{'}\ Q(PW(\downarrow(p_1{\circ}MP(a_1{\circ}q_1){\circ}a_1{\circ}ID{\circ}y_1, 0)))$

Paramodulation B3,1 & R5, $\sigma = \{x \mapsto a_1\}$

$\rightarrow$ R6: $\forall p_1$:'W$\rightarrow$P' $\forall y_1$'P$\rightarrow^{rt}$W' $Q(PW(\downarrow(p_1 \circ MP(a_1 \circ q_1) \circ a_1 \circ y_1, 0)))$

Paramodulation I2,21 & R6, $\sigma = \{x \mapsto a_1, w \mapsto \downarrow(p_1 \circ MP(a_1 \circ q_1), 0), y_1 \mapsto \twoheadrightarrow(a_1, y)\}$

$\rightarrow$ R7: $\forall p_1$:'W$\rightarrow$P' $\forall y$'P$\rightarrow^{rt}$W'

$\neg \downarrow(p_1 \circ MP(a_1 \circ q_1) \circ a_1, 0) \leq \downarrow(p_1 \circ MP(a_1 \circ q_1) \circ y, 0) \vee Q(PW(\downarrow(p_1 \circ MP(a_1 \circ q_1) \circ y, 0)))$

Paramodulation B3,1 & R2, $\sigma = \{x \mapsto q_2, y_2 \mapsto ID\}$

$\rightarrow$ R8: $\forall p_2$:'W$\rightarrow$P' $\neg Q(PW(\downarrow(p_2 \circ a_2 \circ q_2, 0)))$

Paramodulation K1 & R8, $\sigma = \{x \mapsto q_2, w \mapsto \downarrow(p_2 \circ a_2, 0)\}$

$\rightarrow$ R9: $\forall p_2$:'W$\rightarrow$P' $\neg Q(PW(\downarrow(p_2 \circ a_2), 0)))$

Resolution R7,2 & R9, $\sigma = \{p_2 \mapsto p_1 \circ MP(a_1 \circ q_1), y \mapsto a_2\}$

$\rightarrow$ R10: $\forall p_1$:'W$\rightarrow$P'$\neg \downarrow(p_1 \circ MP(a_1 \circ q_1) \circ a_1, 0) \leq \downarrow(p_1 \circ MP(a_1 \circ q_1) \circ a_2, 0)$

A similar sequence where R1 and R2 exchange their role yields

R11: $\forall p_2$:'W$\rightarrow$P' $\neg \downarrow(p_2 \circ MP(a_2 \circ q_2) \circ a_2, 0) \leq \downarrow(p_2 \circ MP(a_2 \circ q_2) \circ a_1, 0)$

2 Paramodulations B3,1 & D12,2, $\sigma = \{x_{B3} \mapsto x_{D12}, z \mapsto ID\}$

$\rightarrow$ R12: $\forall x,y$:'P$\rightarrow^{rt}$W' $\forall p$:'W$\rightarrow$P' $\forall w$:WP $\neg \downarrow(x, w) \leq \downarrow(y, w) \vee \downarrow(MP(x \circ p) \circ x), w) \leq \downarrow(MP(x \circ p) \circ y), w)$

Resolution R10 & R12,2, $\sigma = \{x \mapsto a_1, p \mapsto q_1, y \mapsto a_2, w \mapsto \downarrow(p_1, 0)\}$

$\rightarrow$ R13: $\forall p_1$:'W$\rightarrow$P' $\neg \downarrow(p_1 \circ a_1, 0) \leq \downarrow(p_1 \circ a_2, 0)$

Resolution G4,1 & R13, $\sigma = \{x \mapsto a_1, y \mapsto a_2, w \mapsto \downarrow(p_1, 0)\}$

$\rightarrow$ R14: $\forall p_1$:'W$\rightarrow$P' $\downarrow(p_1 \circ a_2, 0) \leq \downarrow(p_1 \circ a_1, 0)$

Resolution R14 & R11, $\sigma = \{p_1 \mapsto p_2 \circ MP(a_2 \circ q_2)\}$

$\rightarrow$ R15: empty clause. ■

Although this proof looks awfully complicated, most of the steps are simple term rewriting steps which can be handled very well by a demodulation mechanism.

Example 5.3.6 **Axiom G1**

We want to prove $Q\ U^r R \Rightarrow \Diamond^{rt}(Q \wedge R)$

Translation of the negated formula into CL (def. 5.2.2) yields

$\neg(\forall p$:'W$\rightarrow$P' $(\exists a$:'P$\rightarrow^{rt}$W' $(R \wedge \exists y$:'P$\rightarrow^{rt}$W'-a $\leq(p \circ y, p \circ a) \Rightarrow Q))$ $\Rightarrow \exists x$:'W$\rightarrow^{rt}$W$(Q \wedge R))$

The negation normal form is:

$(\forall p$:'W$\rightarrow$P' $\exists a$:'P$\rightarrow^{rt}$W' $(R \wedge \forall y$:'P$\rightarrow^{rt}$W'-a $p \circ y \leq p \circ a \Rightarrow Q))$ $\wedge \forall x$:'W$\rightarrow^{rt}$W$(\neg Q \vee \neg R))$

Translation into OSPL (def. 4.3.2) yields: a:'P$\rightarrow^{rt}$W'

$\forall p$:'W$\rightarrow$P' $R(\downarrow(p \circ a, 0)) \wedge \forall y$:'P$\rightarrow^{rt}$W' $\downarrow(p \circ y, 0) \leq \downarrow(p \circ a, 0) \Rightarrow Q\downarrow(p \circ y, 0)$

$\wedge \forall x$:'W$\rightarrow^{rt}$W$(\neg Q\downarrow(x, 0) \vee \neg R\downarrow(x, 0))$

The clauses are:

G1 $\forall w$:WP $w \leq w$

R1 $\forall p$:'W$\rightarrow$P' $R\downarrow(p \circ a, 0)$

R2 $\forall p$:'W$\rightarrow$P' $\forall y$:'P$\rightarrow^{rt}$W' $\downarrow(p \circ y, 0) \leq \downarrow(p \circ a, 0) \vee Q\downarrow(p \circ y, 0)$

R3 $\forall x$:'W$\rightarrow^{rt}$W $\neg Q\downarrow(x, 0) \vee \neg R\downarrow(x, 0)$

Resolution R1 & R3,2, $\sigma = \{x \mapsto p \circ a\}$ (The sort of p∘a is 'W$\rightarrow^{rt}$W')

$\rightarrow$ R4: $\neg Q\downarrow(p \circ a, 0)$

Resolution R2,2 & R4, $\sigma = \{y \mapsto a\}$

$\rightarrow$ R5: $\neg \downarrow(p \circ a, 0) \leq \downarrow(p \circ a, 0)$

Resolution G1 & R5, $\sigma = \{w \mapsto \downarrow(p \circ a, 0)\}$

$\rightarrow$ R7: empty clause. ■

Example 5.3.7 **Axiom J2**

We want to prove $\forall z{:}D\ \Box^{rt}[\![z]\!]^{r}Q \Rightarrow \forall a{:}D\ |a)\ Q$

Translation of the negated formula into CL yields

$\neg((\forall z{:}D\ \forall x{:}\text{'}W\!\to^{rt}\!W\text{'}\ \forall{\downarrow}(y{:}\text{'}D,W\!\to^{r}\!W\text{'}, z_1{=}z)\ Q) \Rightarrow \forall a{:}D\ \ \forall q{:}\text{'}W\!\to\!P\text{'}\ \wp(BF(a)\circ{+}1)\ Q$

The negation normal form is:

$(\forall z{:}D\ \forall x{:}\text{'}W\!\to^{rt}\!W\text{'}\ \forall{\downarrow}(y{:}\text{'}D,W\!\to^{r}\!W\text{'}, z_1{=}z)\ Q) \wedge \exists a{:}D\ \ \exists q{:}\text{'}W\!\to\!P\text{'}\ \wp(BF(a)\circ{+}1)\ \neg Q$

Translation into OSPL yields: $a{:}D,\ q{:}\text{'}W\!\to\!P\text{'}$

$\forall z{:}D\ \forall x{:}\text{'}W\!\to^{rt}\!W\text{'}\ \forall y{:}\text{'}D,W\!\to^{r}\!W\text{'}\ Q{\downarrow}(x\circ{\downarrow}(y,z), 0) \wedge \neg Q{\downarrow}(q\circ BF(a)\circ{+}1), 0)$

The clauses are:

J2 $\forall l{:}D\ BF(l) \circ {\downarrow}({+}1L, l) = BF(l) \circ {+}1$

R1 $\forall z{:}D\ \forall x{:}\text{'}W\!\to^{rt}\!W\text{'}\ \forall y{:}\text{'}D,W\!\to^{r}\!W\text{'}\ Q{\downarrow}(x\circ{\downarrow}(y,z), 0)$

R2 $\neg Q{\downarrow}(q\circ BF(a)\circ{+}1), 0)$

Paramodulation J2,r & R2, $\sigma = \{l \mapsto a\}$

$\to$ R3: $\neg Q{\downarrow}(q\circ BF(a)\circ{\downarrow}({+}1L, a)), 0)$

Resolution R3 & R1, $\sigma = \{x \mapsto q\circ BF(a),\ y \mapsto {+}1L,\ z \mapsto a\}$

$\to$ R4: empty clause ■

Example 5.3.8 **Axiom J2**

We want to prove $|a)Q \Rightarrow \blacktriangleright\!\langle a\rangle^{\emptyset}Q$ $a{:}D$

Translation of the negated formula into CL yields

$\neg(\forall p{:}\text{'}W\!\to\!P\text{'}\ \wp(BF(a)\circ{+}1)\ Q\ \Rightarrow \forall q{:}\text{'}W\!\to\!P\text{'}\ \exists x{:}\text{'}P\!\to^{rt}\!W\text{'}\ \exists{\downarrow}(y{:}\text{'}D,W\!\to^{\emptyset}\!W\text{'}, z{=}a)\ Q)$

The negation normal form is:

$\forall p{:}\text{'}W\!\to\!P\text{'}\ \wp(BF(a)\circ{+}1)\ Q\ \wedge \exists q{:}\text{'}W\!\to\!P\text{'}\ \forall x{:}\text{'}P\!\to^{rt}\!W\text{'}\ \forall{\downarrow}(y{:}\text{'}D,W\!\to^{\emptyset}\!W\text{'}, z{=}a)\ \neg Q$

Translation into OSPL yields:

$\forall p{:}\text{'}W\!\to\!P\text{'}\ Q{\downarrow}(p\circ BF(a)\circ{+}1), 0)\ \wedge \forall x{:}\text{'}P\!\to^{rt}\!W\text{'}\ \forall y{:}\text{'}D,W\!\to^{\emptyset}\!W\text{'}\ \neg Q{\downarrow}(q\circ x\circ{\downarrow}(y,a), 0)$

$a{:}D,\ q{:}\text{'}W\!\to\!P\text{'}$

The clauses are:

J2 $\forall l{:}D\ BF(l) \circ {\downarrow}({+}1L, l) = BF(l) \circ {+}1.$

R1 $\forall p{:}\text{'}W\!\to\!P\text{'}\ Q{\downarrow}(p\circ BF(a)\circ{+}1), 0)$

R2 $\forall x{:}\text{'}P\!\to^{rt}\!W\text{'}\ \forall y{:}\text{'}D,W\!\to^{\emptyset}\!W\text{'}\ \neg Q{\downarrow}(q\circ x\circ{\downarrow}(y,a), 0)$

Paramodulation J2,r & R2, $\sigma = \{l \mapsto a\}$

$\to$ R3: $\forall p{:}\text{'}W\!\to\!P\text{'}\ Q{\downarrow}(p\circ BF(a)\circ{\downarrow}({+}1L,a)\ ), 0)$

Resolution R3 & R2, $\sigma = \{p \mapsto q,\ x \mapsto BF(a),\ y \mapsto {+}1L\}$

$\to$ R4: empty clause ■

Example 5.3.8 Axiom J3

We want to prove $\Diamond^{t}\langle a\rangle^{\emptyset}Q \Rightarrow \langle a\rangle^{t}Q$ $\quad$ a:D

Translation of the negated formula into CL yields

$\neg(\exists c{:}\text{'}W{\to}^{t}W\text{'}\ \exists{\downarrow}(d{:}\text{'}D,W{\to}^{\emptyset}W\text{'}, z{=}a)\ Q \Rightarrow \exists{\downarrow}(x{:}\text{'}D,W{\to}^{\emptyset}W\text{'}, z{=}a)\ Q$

The negation normal form is:

$\exists c{:}\text{'}W{\to}^{t}W\text{'}\ \exists{\downarrow}(d{:}\text{'}D,W{\to}^{\emptyset}W\text{'}, z{=}a)\ Q \wedge \forall{\downarrow}(x{:}\text{'}D,W{\to}^{\emptyset}W\text{'}, z{=}a)\ \neg Q$

Translation into OSPL yields:

$Q{\downarrow}(c \circ {\downarrow}(d,a)),0) \wedge \forall{\downarrow}(x{:}\text{'}D,W{\to}^{\emptyset}W\text{'}, z{=}a)\ \neg Q{\downarrow}(x,a),0)$

$c{:}\text{'}W{\to}^{t}W\text{'},\ d{:}\text{'}D,W{\to}^{\emptyset}W\text{'}$

The clauses are:

J3 $\forall x{:}\text{'}W{\to}^{rt}W\text{'}\ \forall y{:}\text{'}D,W{\to}^{rt}W\text{'}, \forall l{:}D\ x \circ {\downarrow}(y, l) = {\downarrow}(LT(x, y), l)$

R1 $Q{\downarrow}(c \circ {\downarrow}(d,a)),0)$ $\qquad$ R2 $\forall{\downarrow}(x{:}\text{'}D,W{\to}^{\emptyset}W\text{'}, z{=}a)\ \neg Q{\downarrow}(x,a),0)$

Paramodulation J3,1 & R1, $\quad \sigma = \{x \mapsto c, y \mapsto d, l \mapsto a\}$

$\to$ R3: $Q{\downarrow}({\downarrow}(LT(c, d), a),0)$

Resolution R2 & R3, $\quad \sigma = \{x \mapsto LT(c, d)\}$

$\to$ R4: empty clause.

One further example with an interesting epistemic semantics will be given at the end of section 5.5.

5.4 Induction

In the standard interpretation of MM-Logic with $\mathfrak{R}^{t}$ being the transitive closure of $\mathfrak{R}^{\emptyset}$, paths are isomorphic to natural numbers. Since we have only monadic functions operating on paths, functions like addition or multiplication for example cannot be defined with our restricted syntax. Therefore things might not be as complicated as in number theory. Nevertheless, a first-order axiomatization is not sufficient to obtain a complete calculus. Formulae like

$$P \wedge \Box^{rt}(P \Rightarrow \Box^{\emptyset}P) \Rightarrow \Box^{rt}P \quad \text{or} \quad P \wedge \Box^{rt}(P \Rightarrow \Diamond^{\emptyset}P) \Rightarrow \succ^{rt}P$$

although theorems in MM-Logic are not provable. They are inductive theorems with the usual structure of inductive statements, P is the induction base and $\Box^{rt}(P \Rightarrow \Box^{\emptyset})$ or $\Box^{rt}(P \Rightarrow \Diamond^{\emptyset}P)$ respectively is the induction step. In order to make a theorem prover prove them, an induction mechanism is needed. Unfortunately it turned out that the same problems as they are known from predicate logic inductive theorem proving show up here as well:

➤ The selection problem:
During the search for the proof it has to be decided which formula should be proved by induction. In practical applications such as program verification, the formulae to be proved by induction are usually loop invariants and the like. Thus, selecting the right formula means figuring out the loop invariant, which is usually not a single atom, but a complex formula. Without guidance by the user or some domain specific heuristics this is in general impossible.

➤ Strongly connected with the selection problem is a phenomenon known as the generalization problem. It happens quite frequently that a particular formula $\mathcal{F}$ is not provable by induction, but a proper generalization of $\mathcal{F}$ is. Suppose we have

$\forall x\ Qx$

and $\quad \Box^{rt}(\forall x\ Qx \Rightarrow \Box^{\emptyset}\forall xQx)$

From this $\Box^{rt}\forall xQx$ follows. If, however, we try to prove $\Box^{rt}Qa$, the induction base can be shown very easily but we fail to prove the induction step because the induction hypothesis, "Qa holds in a world $\mathcal{W}$" is too weak. On the other hand, if we generalize $\Box^{rt}Qa$ and try to prove $\Box^{rt}\forall xQx$ there will be no difficulties.

As the following example shows, the generalization phenomenon already occurs in the propositional case. Suppose we have

$\Box^{\emptyset}Q$

and $\Box^{rt}(\Box^{\emptyset}Q \Rightarrow \Box^{\emptyset}\Box^{\emptyset}Q)$

From this $\Box^{rt}\Box^{\emptyset}Q$ follows. But if we try to prove $\Box^{rt}\Diamond^{\emptyset}Q$, the induction step fails again because the induction hypothesis "$\Diamond^{\emptyset}Q$ holds in a world $\mathcal{W}$" is too weak. The more general formula $\Box^{rt}\Box^{\emptyset}Q$, however, is easy to prove.

➤ Proofs of existentially quantified theorems in predicate logic usually require to synthesize their Skolem functions. In the arithmetic example

if	$Q(0)$	
and	$\forall n\ Q(n) \Rightarrow Q(n + f(n))$	
then	$\forall n\ \exists k\ Q(n + k)$	(Skolemized: $\forall n\ Q(n + k(n))$)

you define an auxiliry recursive function $h(0) = 0$, $h(n) := h(n\text{-}1) + f(h(n\text{-}1))$, prove by induction $\forall nQ(h(n))$ and then define the Skolem function $k(n) := h(n) - n$.

Exactly the same procedure is necessary to prove the MM-Logic theorem

if	Q
and	$\Box^{rt}(Q \Rightarrow \blacktriangleright Q)$
then	$\Box^{rt}\Diamond^{t}Q$

which, translated into OSPL, is of the same structure as the arithmetic example:

if	$Q(0)$
and	$\forall x{:}\text{'}W\to^{rt}W\text{'}\ (Q(x(0)) \Rightarrow \forall p{:}\text{'}W\to P\text{'}\ Q((x\circ p\circ f)(0))$
then	$\forall x{:}\text{'}W\to^{rt}W\text{'}\ Q(x\circ k)(0)$

These examples show that induction theorem proving in MM-Logic faces the same problems as induction theorem proving in predicate logic. Although it may turn out that the restriction to monadic context access functions simplifies the technical details, we cannot expect a quick solution to the whole problem. The greate similarities may however allow to apply the methods developed for predicate logic also to MM-Logic. This should be subject to further investigation.

5.5 Interpretations of Multi Modal Logic

Modal Logic formalizes the notion of states and state transitions. What these states describe and what causes the transitions is a matter of interpretation, not of the logic itself. Therefore a few but quite famous interpretations shall briefly be sketched with special emphasis on the interpretation of the operators I chose for MM-logic.

The Temporal Interpretation

The temporal interpretation in general uses only new words for the operators. The states are still abstract "states of the world", whatever the "world" is, only the accessibility relation is interpreted as *temporal* development of the "world". That means, the structure of the accessibility relation is used to model time. A linear structure models a straight flow of time, a branching structure models alternative futures. The structure may be discrete, modeling distinguished time ticks as for example the internal clock in computers, or it may be dense as in the real world (neglecting possible quantum effects). The modal operators can relate the current world, i.e. "now", with worlds in the future or in the past.

In MM-Logic we have branching accessibility relations based on the discrete relation $\Re^{\emptyset}$, i.e. we model alternative futures with discrete time steps, however with an incomplete calculus with respect to the particular aspect of discreteness. Time structures which are not discrete can easily be modeled by considering only the operators related to the transitive accessibility relation and discarding the corresponding discreteness axioms. Dense structures can be obtained with the additional axiom

$$\forall x{:}`P\rightarrow^{t}W' \; \exists y{:}`P\rightarrow^{t}W' \; \forall w{:}WP \;\; y(w) < x(w).$$

The seriality assumption means that there is always a future and the constant-domain assumption means that the world itself is static, no objects appear or disappear.

The temporal interpretation of our operators is:

$\Box^{\emptyset}$ always in the next future	$\Diamond^{\emptyset}$ sometime in the next future
$\Box^{r}$ now and always in the next future	$\Diamond^{r}$ now or sometime in the next future
$\Box^{t}$ always in the future	$\Diamond^{t}$ sometime in the future
$\Box^{rt}$ now and always in the next future	$\Diamond^{rt}$ now or sometime in the future
◗ eventually	➔ possibly henceforth
$\forall U$, $\forall U^{r}$ always until	(we have no past operators)
$\exists U$, $\exists U^{r}$ possibly until	

The indexed operators can't be interpreted properly on this general level. Since they are related to the labels of the transitions we have to interpret the labels first.

The Process Interpretation

In a special temporal interpretation, the worlds are related to the internal states of processes, software, hardware or whatsoever, which may be in distinguished states and may perform certain atomic actions causing transitions into successor states. The process variables can be represented by flexible constant symbols. Functions and predicates which may change their definition (as for example in Lisp) can be represented by flexible function and predicate symbols.

Since we consider branching time, the processes' actions may be nondeterministic. If the processes are software modules, a possible worlds structure therefore describes their complete computation tree, and an actual computation is represented as a path through the possible worlds structure.

Now we can give a concrete interpretation to the labels of the transitions. If we have n processes running in parallel, we choose as label the index of the process that causes that particular transition. The seriality of labeled $\Re^{\emptyset}$-transitions however means that at each state (world) each process performs an action, and this is unrealistic. Therefore only the operators corresponding to the transitive relation which require the existence of some labeled $\Re^{\emptyset}$-transitions in the future make sense in a process interpretation. The interpretation of these operators are:

$[\![n]\!]^t$	always after the n-th process´ action.
$<n>^t$	sometimes (in the branching computation tree) after the n-th process´ action.
$[\![n]\!]^{rt}$	now and always after the n-th process´ action.
$<n>^{rt}$	now or sometimes (in the branching computation tree) after the n-th process´ action.
$\mid n)$	immediately after the n-th process´ next action.

The incompleteness of the axiomatization of the $\Re^t$-relation, however, means that we can not yet prove inductive properties like loop invariants.
The seriality assumption means that there are no deadlocks. All processes will eventually perform a transition. The constant-domain assumption means that we cannot model dynamic creation and deletion of processes in a natural way, without special tricks.

The Interpretation as Action Logic

This interpretation is slightly different to the previous one. The difference is that this time we do not label the transitions with processes that cause actions, but with actions themselves. We can for example write $\forall$x:Colour $[\![$paint(wall, x)$]\!]^t$ color(wall, x) with the intended meaning: whenever the wall is painted, it will have that colour. (‘Colour’ is a sort symbol and ‘color’ is a predicate.) The interpretation in terms of possible worlds is: in each state of the world that is created by a ‘paint(wall, x)’-action (transition), color(wall, x) will hold. This statement is an example for the use of the transitive $[\![\ldots]\!]^t$-operator to express invariants. The statement $[\![$lose(Tom,1000\$)$]\!]^{\emptyset}$ bankrupt(Tom), saying that when Tom gambles away another 1000\$ he will be bankrupt, however, is no invariant. It may be true now and false tomorrow (when his rich uncle has died). Therefore the $[\![\ldots]\!]^{\emptyset}$-operator is appropriate (but without the assumption about seriality of labeled transitions because Tom needs not gamble at all).

The Epistemic Interpretation

Hintikka originally had the idea of formalizing the propositional attitude of belief with possible worlds [Hintikka 62]. The basic concept is that the propositions of an actor´s (say A) belief are represented as a set of worlds, compatible with A´s beliefs. Any member of this set is, according to the way A thinks, a candidate for the real world, that is

A beliefs $\mathcal{F}$ if and only if for all $\mathcal{W} \in$ possible-worlds(A), $\mathcal{F}$ is true in $\mathcal{W}$.

Levesque, Halpern and Moses, Konolige and others have developed this idea to a formal logic with a tableau based deduction calculus [Levesque 84], [Konolige 86], [Halpern&Moses 85].

In epistemic logics the indexed modal operators $[\![\ldots]\!]$ are used to express belief. For example $[\![A]\!]\mathcal{F}$ is interpreted: A believes, $\mathcal{F}$ is true, whereas $<A>\mathcal{F}$ means: A thinks, $\mathcal{F}$ might be possible. The operators related to the different accessibiltiy relations are interpreted as follows:

$[\![A]\!]^{\emptyset}\mathcal{F}$	A beliefs $\mathcal{F}$, it might however be false in the “real” world.
$[\![A]\!]^{r}\mathcal{F}$	A beliefs $\mathcal{F}$ and $\mathcal{F}$ it is actually true in the real world (A is an expert for $\mathcal{F}$)
$[\![A]\!]^{t}\mathcal{F}$	A beliefs $\mathcal{F}$ and he beliefs that he beliefs it etc. (introspection)
$[\![A]\!]^{rt}\mathcal{F}$	A beliefs $\mathcal{F}$ and he beliefs that he beliefs it etc. and $\mathcal{F}$ is actually true.

(In order to enforce that not only the last $\Re^{\emptyset}$-transition in a labeled $\Re^t$-transition matters for the interpretation of the transitive operators, but the lables of all transitions in the sequence have to be the same, simply discard axiom J3 in def. 5.2.3. This is more appropriate for the epistemic interpretation of these operators.)

MM-Logic has some features which may make it quite useful as an epistemic logic:

- It allows the simultaneous use of the different belief operators.
- It allows arbitrary non-ground terms as representation for actors.
  For example the, "common knowledge" operator is simply $\forall x{:}\mathrm{Actor}\ [\![x]\!]^{\emptyset}\ldots$
- With a slight extension it is even possible to have a very selective "implicit knowledge" operator.
  All you have to do is to incorporate a special unification algorithm for a particular kind of sets which "unifies" two sets by simply uniting them.
  Then we can deduce from $[\![A]\!]P$ and

$$\frac{[\![A]\!]P \qquad [\![B]\!](P \Rightarrow Q)}{[\![A \cup B]\!]Q}$$

  i.e. when A and B join their knowlege, they can deduce Q form A´s knowledege of P and B´s knowldege of $P \Rightarrow Q$. This unification rule realizes the axiom scheme $[\![X]\!]\mathcal{F} \Rightarrow [\![X \cup Y]\!]\mathcal{F}$ saying when X joins his knowledge with Y then both together know at least what X knows.

What MM-Logic cannot model is:

- Inconsistent knowledge.
  Due to the seriality assumption, from every world there is for every actor a consistent world, i.e. his knowledge is always consistent.
- Restrictions for the tautology "$[\![X]\!]$true" which is the unrealistic assumption that everybody knows all logical truth (the famous omniscience problem).With the current version of OSPL as target logic, there is no way to switch for example to weaker non normal S2 based systems where "$[\![X]\!]$true" holds only for predicate logic truth.
- In the epistemic interpretation of MM-Logic, every actor knows all consequences of his knowledge (deductive closure property). No restrictions to the deductive closure are possible so far.

There is a famous example from McCarthy, the *wise man puzzle*, that has been used to test the representation ability of formalisms for knowledge and belief. As a final example we give an axiomatization of the wise man puzzle and a proof in MM-Logic. Its traditional form is:

A certain king wishes to determine which of his three wise men is the wisest. He arranges them in a circle so that they can see and hear each other and tells them that he will put a white or black spot on each of their foreheads but at least one spot will be white. In fact all three spots are white. He then offers his favor to the one who first tells him the color of his spot. After a while, the wisest announces that his spot is white. How does he know?

(Actually the information that all three spots are white is not necessary to solve the puzzle.)

The solution involves the wisest man reasoning about what his colleagues know and don´t know from observations and the king´s announcement.

To axiomatize this puzzle in epistemic logic, assume the three wise man are A, B and C and C is the wisest.

First of all we need the three formulae:

C1: $A \neq B$

C2: $A \neq C$

C3: $B \neq C$

and assume the symmetry of the $\neq$-predicate.

At least one of them has a white spot and everybody knows that everybody else knows that his colleagues know this.

C4: $\forall S, S', S'': [\![S]\!]\,[\![S']\!]\,[\![S'']\!]\, W(A) \vee W(B) \vee W(C)$

(W(S) means S has a white spot. The $[\![\ldots]\!]$-operator is actually the $[\![\ldots]\!]^{\varphi}$-operator.)

The three men can see each other and they know this. Therefore whenever one of them has a white or black spot, he knows that his colleagues know this and he knows also that his colleagues know this from each other.

C5: $\forall S,S':\ S \neq S' \Rightarrow [\![S]\!]\,(\neg W(S) \Rightarrow [\![S']\!]\neg W(S))$

C6: $\forall S,S',S''\ S \neq S' \wedge S \neq S'' \wedge S' \neq S'':\ \Rightarrow [\![S]\!]\,[\![S']\!]\,(\neg W(S) \Rightarrow [\![S'']\!]\neg W(S))$

C7: $\forall S,S',S''\ S \neq S' \wedge S \neq S'' \wedge S' \neq S'':\ \Rightarrow [\![S]\!]\,[\![S']\!]\,(\neg W(S') \Rightarrow [\![S'']\!]\neg W(S'))$

(We give only the minimum number of axioms which are necessary for the proof.)

They can hear each other and they know this. B did not say anything, therefore C knows that B does not know the colour of his own spot.

C8: $[\![C]\!]\neg\,[\![B]\!]\,W(B) \qquad (\Leftrightarrow [\![C]\!] \langle B\rangle \neg W(B))$

C knows that B knows that A does not know the colour of his spot.

C9: $[\![C]\!]\,[\![B]\!]\neg\,[\![A]\!]\,W(A) \quad (\Leftrightarrow [\![C]\!]\,[\![B]\!] \langle A\rangle \neg W(A))$.

We translate the formulae into OSPL syntax:

The sort of the variables in lowercase symbols is '$W \rightarrow^{\varphi} W$'.

To make the formula more readable we use second order syntax and drop the $\circ$ and $\downarrow$-function and the 0 sign writing terms $\downarrow(x\circ\ldots\circ z, 0)$ in simple brackets $[x\ldots z]$.

C1: $A \neq B$ C2: $A \neq C$ C3: $B \neq C$

C4: $\forall S,u,S',u',S'',u'':\ W([u(S)\,u'(S')\,u''(S'')], A) \vee W([u(S)\,u'(S')\,u''(S'')], B) \vee W([u(S)\,u'(S')\,u''(S'')], C)$

C5: $\forall S,u, S',u':\ S = S' \vee W([u(S)], S) \vee \neg W([u(S)\,u'(S')], S)$

C6: $\forall S,u, S',u', S'',u'':\ S = S' \vee S = S'' \vee S' = S'' \vee W([u(S)\,u'(S')], S) \vee \neg W([u(S)\,u'(S')\,u''(S'')], S))$

C7: $\forall S,u, S',u', S'',u'':\ S = S' \vee S = S'' \vee S' = S'' \vee W([u(S)\,u'(S')], S') \vee \neg W([u(S)\,u'(S')\,u''(S'')], S'))$

C8: $\forall u\ \neg W([u(C)\,g(B)], B)$

C9: $\forall u,v\ \neg W([u(C)\,v(B)\,h(A)], A)$

A deduction of the fact that C knows the colour of his own spot, i.e. $[\![C]\!]W(C)$ is now a trivial exercise for any resolution theorem prover. The following UR-proof was found by our system [Eisinger&Ohlbach 86]:

C1,C2,C3,C7,C8	$\rightarrow$ R1: $\forall u,u''\ \neg W([u(C)\,g(B)\,u''(A)], B)$	$(\Leftrightarrow [\![C]\!]\langle B\rangle [\![A]\!]\neg W(B))$	
C9, R1,C4	$\rightarrow$ R2: $\forall u\ W([u(C)\,g(B)\,h(A)], C)$	$(\Leftrightarrow [\![C]\!]\langle B\rangle\langle A\rangle W(C))$	
C1,C2,C3,R2,C6	$\rightarrow$ R3: $\forall u\ W([u(C)\,g(B)], C)$	$(\Leftrightarrow [\![C]\!]\langle B\rangle W(C))$	
C3,R3,C5	$\rightarrow$ R4: $\forall u\ W([u(C)], C)$	$(\Leftrightarrow [\![C]\!] W(C))$	■

Actually this example is so simple that none of the axioms for the accessibility relation (def. 5.2.3) are necessary. A standard unsorted resolution calculus with Robinson unification is sufficient.

Chapter Six

Summary

A method for translating formulae of a large class of first-order logics with possible worlds semantics into predicate logic has been developed. The method allows for theorem proving by translation - into predicate logic - and refutation - with predicate logic resolution and paramodu- lation. The basic idea of the translation is to transform the information contained in a sequence of nested nonclassical operators into one term representing explicitly the whole path through the possible worlds structure from the initial world to the actual world that is used to interpret terms and atoms. These "context terms" are attached as additional arguments to the terms and atoms. Since they are ordinary terms, predicate logic unification can treat them in the usual way. That means, one resolution or paramodulation step has, via unification, the whole relevant part of the possible worlds structure to its disposal and therefore needs not jump shortsightedly from one world to another, as this is for example the case in some tableaux or sequent calculi [Fitting 72,83]. Furthermore, since the operators are represented as terms containing variables, their unification can be seen as the computation of the "most general world", again an advantage over calculi which can handle only explicitly generated worlds one by one. Since the "target logic" for the translation is order-sorted predicate logic (OSPL) with a fully developed resolution and paramodulation calculus, the "source logics" may also be first-order, order-sorted with built-in equality.

Context Logic (CL) has been developed as an intermediate logic between the source logics and predicate logic because there are a number of translation operations common to all source logics which can be handled by this method, and these operations can be comprised in the translation from CL to OSPL. The translation from the source logic into CL is almost a one to one translation of the model theoretic semantics of its operators, whereas the translation from CL to OSPL contains the shift of information from the operator level to the term level.

In order to demonstrate that using Context Logic simplifies the task of designing a proof theory for a logic considerably, we have applied the method to a quite complex first-order order-sorted multi modal logic with built-in equality. It is based on the modal logics D, T, D4 and S4 and includes CTL. It contains several kinds of multiplicities. We allow modal operators corresponding to a basic accessibility relation to occur simultaneously with operators corresponding to its reflexive, transitive and reflexive transitive closure. Furthermore we have indexed operators which refer to labeled transitions in the possible worlds structure. The indices can be arbitrary - possibly nonground - terms. Finally we have an 'eventually' operator (on each path there is a world such that ...), an indexed eventually operator and some 'until' operators. The logic is quite expressive and can serve in various applications as temporal logic, process logic, action logic or epistemic logic.

The limits of the translation method based on Context Logic are:
- the semantical structure behind the source logic must be first-order axiomatizable, otherwise the resulting calculus is not complete.
- We consider only constant-domain interpretations.
- From each context (world) there must be an accessible world (seriality).

The last two points are due to the fact that OSPL as target logic cannot handle partial functions. Once it has been extended to handle partial functions whose domain have to be deduced during the proof, CL can be extended too and we can allow nonserial varying-domain logics as source logics.

The whole translation idea for logics and CL in particular is a natural extension of basic work done for classical modal logics. It began with calculi using labeled formulae where the labels are either integers [Chan 87] or terms denoting worlds [Wallen 87]. The labels were not made part of the syntax, but used to guide the proof system.

After that, different groups came up with the idea to make the terms part of the syntax, thus translating modal logic syntax into predicate logic syntax [Enjalbert&Auffray 89], [Fariñas & Herzig 88], [Ohlbach 88]. Context Logic now systematizes these efforts and supports the designer of a logic in developing a proof theory by translation and refutation.

Future Work

To improve Context Logic itself, first of all OSPL should be extended to handle partial functions whose domain is unknown at the beginning. This is necessary to deal with nonserial accessibility relations whose context access functions are partial and to allow varying-domain interpretations where terms may be undefined in some worlds. For modal logics with nonserial accessibility relations I have developed in [Ohlbach 88] a resolution calculus where this problem is solved using theory resolution rules [Stickel 85] which take care of partially defined functions. This could be a guideline to do it for OSPL in general.

How to extend CL for dealing with semantical structures which are not first-order axiomatizable is not so obvious. As the modal logic G shows, there are Kripke structures which are not first-order axiomatizable, but which nevertheless admit a complete logic. The axiomatization of its Kripke structure must lie in an "inoffensive" second-order fragment. Identifying such fragments and realizing them as special theory resolution rules is surely a challenging task.

Of course I am anxious to see the Context Logic translation method applied to other logics. Temporal logics based on intervals should be a promising candidate.

The calculus for MM-Logic as it is can run on a predicate logic theorem prover with built-in equality handling and overloaded sort declarations. To make it really efficient, however, the clauses axiomatizing the possible worlds structure (def. 5.3.1) should be turned into unification algorithms. Since the sort structure separates them form user defined equations, it is possible to consider them separately without the danger of interferences with user defined axioms.

In many applications not all of the MM-Logic operators will be necessary. In this case some of the clauses in def. 5.3.1 become superfluous and the corresponding unification algorithms should become simpler and more efficient. Furthermore, as it turned out to happen for modal logics [Ohlbach 88], it may be possible to identify syntactic invariants which hold initially for the translated formulae and which are preserved during the deduction. Exploiting these invariants may improve the unification algorithms considerably. This kind of improvements depend on the particular application and should be investigated separately.
A much more expensive enterprise is the incorporation of induction mechanisms into MM-Logic theorem proving. As we have seen in section 5.4, induction theorem proving is neccessary to approximate the transitive closure of the basic accessibility relation. This is of particular importance for the application of MM-Logic as process logic. It seems that the translation into OSPL at least provides the syntactic basis for transferring the methods developed for predicate logic induction theorem proving to MM-Logic.

Acknowledgements

I wish to thank Andreas Nonnengart who contributed to the development of MM-Logic. He found all counterexamples to its earlier incomplete versions. I hope he won't find any more. I am also grateful to Michael McRobbie who invited me for a visit to the Automated Reasoning Project at the Australian National University. Freed from the daily duties at home and not bound to new ones I was able to develop the formalism and to write this report during my stay. I doubt that at home I could have found the time and concentration to finish it at all.

References

Boyer&Moore 79	R.S. Boyer, J.S. Moore: *A Computational Logic.* Academic Press 1979.
Chan 87	M. Chan. *The Recursive Resolution Method.* New Generation Computing, 5 pp. 155-183, 1987.
Chang&Lee 73	C.-L.Chang, R.C.-T. Lee. *Symbolic Logic and Mechanical Theorem Proving.* Science and Applied Mathematics Series (ed. W. Rheinboldt), Academic Press, New York, 1973.
Clarke&Emerson 81	M.C. Clarke, E.A. Emerson. *Design and Synthesis of Synchronization Skeletons using Branching Time Temporal Logic.* Lecture Notes in Computer Science 131, Springer Verlag, New York, 1981, pp. 52-71.
Eisinger&Ohlbach 86	N. Eisinger, H.J.Ohlbach. *The Markgraf Karl Refutation Procedure.* Proc. of 8th Conference on Automated Deduction, pp. 682-683, 1986.
Enjalbert&Auffray 89	P. Enjalbert, Y. Auffray. *Modal Theorem Proving: An Equational Viewpoint* Submitted to IJCAI 89.
Fariñas&Herzig 88	L.Fariñas del Cerro, A.Herzig *Quantified Modal Logic and Unification Theory* Langages et Systèmes Informatique, Université Paul Sabatier, Toulouse. Rapport LSI n° 293, jan. 1988. See also L. Fariñas del Cerro, A. Herzig *Linear Modal Deductions.* Proc. of 9th Conference on Automated Deduction, pp. 487-499, 1988.
Fitting 72	M.C. Fitting. *Tableau methods of proof for modal logics.* Notre Dame Journal of Formal Logic, XIII:237-247,1972.
Fitting 83	M.C. Fitting. *Proof methods for modal and intuitionistic logics.* Vol. 169 of Synthese Library, D. Reidel Publishing Company, 1983.
Grätzer 79	G. Grätzer. *Universal Algebra.* Springer Verlag (1979).
Halpern&Moses 85	J.Y. Halpern and Y. Moses. *A guide to modal logics of knowledge and belief:* preliminary draft. In Proc. of 9th IJCAI, pp 479-490, 1985.
Hughes&Cresswell 68	G.E.Hughes, M.J.Cresswell. *An Introduction to Modal Logics.* Methuen &Co., London, 1986.
Hintikka 62	J. Hintikka. *Knowledge and Belief.* Cornell University Press, Ithaca, New York, 1962.
Konolige 86	K.Konolige. *A Deduction Model of Belief and its Logics.* Research Notes in Artificial Intelligence, Pitman, London, 1986.
Kripke 59	S. Kripke. *A Completeness Theorem in Modal Logic.* J. of Symbolic Logic, Vol 24, 1959, pp 1-14.
Kripke 63	S. Kripke. *Semantical analysis of modal logic I, normal propositional calculi.* Zeitschrift für mathematische Logik und Grundlagen der Mathematik, Vol. 9, 1963, pp.67-96.
Levesque 84	H.J. Levesque. *A logic of knowledge and active belief.* Proc. of American Association of Artificial Intelligence, University of Texas, Austin 1984.
Loveland 78	D. Loveland: *Automated Theorem Proving: A Logical Basis.* Fundamental Studies in Computer Science, Vol. 6, North-Holland, New York 1978.
Moore 80	R.C. Moore. *Reasoning about Knowledge and Action.* PhD Thesis, MIT, Cambridge 1980.
Ohlbach 88	H.J. Ohlbach. *A Resolution Calculus for Modal Logics* Thesis, FB. Informatik, University of Kaiserslautern, 1988.
Robinson 65	J.A. Robinson. *A Machine Oriented Logic Based on the Resolution Principle* J.ACM, Vol. 12, No 1, 1965, 23-41.
Robinson & Wos 69	Robinson, G., Wos, L. *Paramodulation and theorem provcing in first order theories with equality.* Machine Intelligence 4, American Elsevier, New York, pp. 135-150, 1969.
Schmidt-Schauß 85	Schmidt-Schauß, M. *A Many-Sorted Calculus with Polymorphic Functions Based on Resolution and Paramodulation.* Proc. of 9th IJCAI, Los Angeles, 1985, 1162-1168.
Schmidt-Schauß 88	Schmidt-Schauß, M. *Computational aspects of an order-sorted logic with term declarations.* Thesis, FB. Informatik, University of Kaiserslautern, 1988.
Smullyan 68	R.M. Smullyan. *First Order Logic,* Springer Verlag, Berlin 1968.
Stickel 85	M. Stickel. *Automated Deduction by Theory Resolution.* Journal of Automated Reasoning Vol. 1, No. 4, 1985, pp 333-356.
Wallen 87	L.A.Wallen. *Matrix proof methods for modal logics.* In Proc. of 10th IJCAI, 1987.
Walther 87	C. Walther: *A Many-sorted Calculus Based on Resolution and Paramodulation.* Research Notes in Artifical Intelligence, Pitman Ltd., London, M. Kaufmann Inc., Los Altos, 1987.