Bringing them all Together

Carlos Areces[1]
Patrick Blackburn[2]

[1]ILLC, University of Amsterdam
Plantage Muidergracht 24, 1018 TV Amsterdam, The Netherlands
Email: `carlos@science.uva.nl`
[2]INRIA, Lorraine
615, rue du Jardin Botanique, 54602 Villers lès Nancy Cedex, France
Email: `patrick@aplog.org`

[. . .] No way to say *warm* in French. There was only *hot* and *tepid*. If there's no word for it, how do you think about it? [. . .] Imagine, in Spanish having to assign a gender to every object: dog, table, tree, can-opener. Imagine, in Hungarian, not being able to assign a gender to anything: *he*, *she*, *it* all the same word. Thou art my friend, but you are my king; thus the distinctions of Elizabeth the First's English. But with some oriental languages, which all but dispense with gender and number, you are my friend, *you* are my parent, and YOU are my priest, and *YOU* are my king, and **You** are my servant, and ***You*** are my servant whom I'm going to fire tomorrow if ***You*** don't watch it, and **YOU** are my king whose policies I totally disagree with and have sawdust in **YOUR** head instead of brains, **YOUR** highness, and **YOU** may be my friend, but I'm still gonna smack **YOU** up side the head if **YOU** ever say that to me again: and who the hell are you anyway. . . ?

Babel-17
Samuel R. Delany

1 What are Hybrid Logics?

Hybrid logics are modal logics and — at least, if the authors of this editorial had their way — vice-versa. Strictly speaking, not all modal logics are hybrid, but certainly any modal logics can be *hybridized*, and in our view many of them should be. What is it to hybridize a modal logic? To answer this properly we need to step back a little and discuss recent trends in modal logic.

Starting from the beginning, the new (sometimes call Amsterdam-style) perspective on modal logic, considers modal languages as general tools for talking about relational structures (for an up-to-date presentation of modal logic from this perspective, see [14]). A Kripke model for a propositional modal language is simply a set of points on which various relations have been defined, together with an assignment of atomic information — so Kripke models are just relational structures, the kinds of structures used to interpret first- and second-order classical languages. Relational structures are a useful tool in many disciplines: computer scientists can view labeled transition systems as relational structures, AI researchers can view various pictures of time as relational structures, description logicians can view networks of individuals as relational structures, and philosophers can view 'possible worlds' and the links between them in these terms. But none of these interpretations is privileged — and from the new perspective, that's the whole point. Relational structures are a fundamental modeling tool, and one reason why modal logic is so widely applicable is simply that it can be used to reason about whatever relational structures researchers find interesting.

But (according to the new perspective) there is a second reason why modal logic is so often the logic of choice. In essence, classical modal operators like $\Diamond$ and $\Box$ are 'macros' which help us uncover interesting fragments of first- and second-order classical logics. A unary diamond $\Diamond$ bundles a relatively simple form of classical quantification ("look for the information you are interested in at some accessible point") into an even simpler operator notation. The Until operator used in temporal applications bundles up a more complex $\exists\forall$ quantification pattern into a simple binary operator format. The $\langle\pi^*\rangle$ of Propositional Dynamic Logic ("look for the information you are interested by making a finite number of π transitions") bundles up quantification over the reflexive transitive closure of a relation into a unary operator. In short, the game of modal logic is about finding flexible and malleable operators which, when combined in different ways, yield well-behaved and useful fragments of classical logic.

The new perspective has had two positive effects. First, it has enriched our theoretical understanding of what modal logic is, for in order to fully understand these extensions (for example, to learn which fragments of classical logic they correspond too) new tools such as the Standard Translation and bisimulations are needed (see [14] for a full discussion of these concepts). Second, and just as importantly, it has encouraged modal logicians to think of themselves as 'logic engineers,' whose task is to craft logics to fit particular applications, and this has lead to the development of many new 'extended modal logics.'

It's at this point that hybrid logic comes in. Ordinary modal logics (even those with Until, or with the full apparatus of Propositional Dynamic Logic) have an obvious limitation: they lack a mechanism for referring to the points in models. And for many applications this is a genuine weakness: when reasoning about time, we often want to reason about what happens at a particular instant or interval, and when reasoning about terminologies, we often want to reason about how they apply to particular individuals. Ordinary modal logics don't deliver the goods here, and as logic engineers it is our job to investigate the situation, and add what is needed to complete the picture. This is the road that leads to hybrid logic, for one way to define "hybridization" is as the enrichment of ordinary modal logic with mechanisms for naming and reasoning about individual elements in the domain of a model.

2 Hybridizing Basic Modal Logic

To make the discussion concrete, let's see what is involved in hybridizing the basic modal language (that is, a propositional modal language containing only a single diamond $\Diamond$). Along the way we will meet many of the tools used by the contributors to this special issue: *nominals* (and *state variables*), *satisfaction operators*, the *global* *modality*, and *binders* (in particular, the $\downarrow$ *binder*).

Let's first recall the syntax and semantics of ordinary propositional modal logic. Given a set PROP of propositional variables p, q, r, and so on, we build formulas over PROP as follows:

$$\mathsf{WFF} \ := \ p \mid \neg\varphi \mid \varphi \wedge \psi \mid \Diamond\varphi.$$

Other boolean connectives can be defined in the usual way, and $\Box\varphi$ is $\neg\Diamond\neg\varphi$.

Such a language is interpreted on models $M = (W, R, V)$. Here W is a non-empty set of points (or 'times,' or 'states,' or 'worlds,' or 'individuals,' depending on the application we have in mind), R is a binary relation on W, and V is a function with domain PROP and range 2^W (that is, V assigns to each propositional symbol the set of points at which it is true). The pair (W, R) is usually called a frame, and V is called a valuation. Given such a model, the satisfaction definition for our language is as follows:

$$\begin{array}{rcl} M, w \models p & \text{iff} & w \in V(p) \\ M, w \models \neg\varphi & \text{iff} & M, w \not\models \varphi \\ M, w \models \varphi \wedge \psi & \text{iff} & M, w \models \varphi \ \&\ M, w \models \psi \\ M, w \models \Diamond\varphi & \text{iff} & \exists u(wRu \ \&\ M, u \models \varphi). \end{array}$$

If φ is evaluated at a point w in a model M, we say that w is the "point of evaluation" or the "current point." If $M, w \models \varphi$, we say φ is satisfied at M in w. If $(W, R, V), w \models \varphi$ satisfies φ for any choice of V or w, we say φ is valid on the frame (W, R).

Nominals and Satisfaction Operators

As the satisfaction definition just given makes clear, while $\Diamond$ is an elegant tool for quantifying over R-accessible points, the basic modal language offers us no tools for naming or reasoning about the points in W. Let's put this right by defining the *basic hybrid language.*

Let NOM be a set distinct from PROP. The elements of NOM are called *nominals* and are typically written i, j, k, and so on. We build formulas over NOM and PROP as follows:

$$\mathsf{WFF} \ := i \mid p \mid \neg\varphi \mid \varphi \wedge \psi \mid \Diamond\varphi \mid @_i\varphi.$$

Nominals are the principal hybrid mechanism for referring to points, thus they play the role played by terms in classical logic. But note: *nominals are formulas, not terms.* Further, note that nominals can occur as subscripts to the @ symbol. Such a combination — for example, $@_i$ — is called a *satisfaction operator.*

The interpretation of this language is straightforward. The key step is to redefine what we mean by a valuation. We say that a (hybrid) valuation is a function V with domain $\mathsf{NOM} \cup \mathsf{PROP}$ and range 2^W such that for all $i \in \mathsf{NOM}$, $V(i)$ is a *singleton* set. That is, whereas ordinary propositional variables can be true at any number of points in a model, nominals are true at precisely one point in any model. They 'name' this point by being true there and nowhere else. We often call the unique point in $V(i)$ the *denotation* of i.

With that in place, add now the following two clauses to the satisfaction definition:

$$\begin{array}{rcl} M, w \models i & \text{iff} & w \text{ is the denotation of } i. \\ M, w \models @_i\varphi & \text{iff} & M, u \models \varphi, \text{ where } u \text{ is the denotation of } i. \end{array}$$

That is, nominals are true at a unique point in any model (namely their denotation), and the satisfaction operators $@_i$ shifts the point of evaluation to the denotation of i. So $@_i\varphi$ says: "φ is satisfied at the point named by i."

Note that the formula prefixed by a satisfaction operator can itself be a nominal. For example, $@_ij$ is a well formed formula, and it has a useful meaning: it asserts that the nominal j is true at the point named by i, or to put it more simply, that i and j name the same point. Thus satisfaction operators give us a *modal theory of state equality.* Moreover, note that the formula $@_i\Diamond j$ means that the point named by j is a successor of the point named by i, so satisfaction operators also give us a *modal theory of state succession.*

From a logic engineering perspective, it should be clear that hybridization has added something useful. For example, if we were working in a temporal application (that is, if we think of the set W as a set of times, and the relation R as the relation of temporal precedence) then the formula $@_i\varphi$ says that the information φ holds at the time named by i — in short, it performs the same role as James Allen's [1] **Holds** predicate, but in a modal language. And if we think of W as a set of individuals, and R as a binary role (that is, if we are reasoning about terminologies) then $@_i\varphi$ is what a description logician would call an A-Box assertion.

What is less obvious is that far from having damaged the underlying modal logic, the hybridization we have just witnessed has arguably improved its logical behaviour. To give four examples:

1. *More expressivity over frames.* Ordinary modal languages have some surprising weaknesses when it comes to expressing properties of frames. For example, in ordinary modal languages there is no formula which defines irreflexivity (that is, in ordinary modal logic there is no formula valid on precisely those frames where $\forall w \neg wRw$). But the hybrid formula $@_i\neg\Diamond i$ defines this properly. Similarly, neither antisymmetry, asymmetry, nor discreteness can be defined in ordinary modal logic, but they can all be defined in hybrid logic. See [22, 11] for further discussion.

2. *No computational cost.* Often, adding nominals and satisfaction operators does not raise the complexity of the satisfaction problem. For example, the satisfaction problem of the basic modal language is PSPACE complete, and we remain in PSPACE if we add nominals and satisfaction operators. And the satisfaction problem for Propositional Dynamic Logic is EXPTIME-complete, and we remain in EXPTIME is we add nominals, satisfaction operators, and even the global modality. See [4] for further discussion.

3. *General completeness results.* One of the oldest themes in hybrid logic is that hybridization leads to simpler and more general completeness results. In a nutshell, this is because the presence of nominals and satisfaction operators makes it possible to combine the first-order Henkin construction with the modal canonical model construction (see Chapter 7.3 of [14]). Although ordinary modal logic has general completeness results (notably the Sahlqvist Theorem), these are typically complex to state and difficult to prove. In hybrid logic the situation is far simpler: *any formula which contains no propositional variable is guaranteed to be complete with respect to the class of frames it defines.* For example, $@_i\neg\Diamond i$

defines irreflexivity, and it contains no propositional variables, only nominals. So if we add it as an axiom to a suitable base logic, it is guaranteed to be complete with respect to the class of irreflexive frames.

4. *Proof-theoretical simplicity.* Ordinary modal logics are hard to work with proof-theoretically, for in general there is no simple way to get at the information under the scope of a modality. Again, nominals and satisfaction operators remove this difficulty. For example, suppose we are carrying out a tableau proof and we know that $@_i\Diamond\varphi$. By introducing a new nominal (say j) onto the tableau, we can decompose this information into $@_i\Diamond j$ and $@_j\varphi$, thus pulling the φ out from under the scope of the $\Diamond$. In short: we can carry out tableau proofs by constructing a modal theory of state succession. Another way of looking at it is that we are using a form of labeled deduction (see [20]) — but the labeling apparatus used here (namely nominals and satisfaction operators) belongs to the hybrid object language, hence we have *internalized* labeled deduction. Two contributions to this special issue (namely Seligman's, and that of Areces, de Nivelle and de Rijke) explore the proof theoretical ramifications of hybridization in detail.

Summing up: from both a theoretical and a logic engineering perspective, hybridization has much to offer. But we haven't yet told you everything you need to know. Two more topics remain: the global modality and binders.

The Global Modality

The global modality A (often called the universal modality [25]) can be quickly dealt with: $A\varphi$ means "φ is true at all points in the model." Thus $\neg A\neg\varphi$, which is usually abbreviated to $E\varphi$, means "φ is true at some point in the model." The modality is useful for a number of purposes — for example, to enforce global constraints on terminological definitions. However the operator was first isolated in the hybrid logical tradition, and for a rather different reason: it can be used instead of satisfaction operators, for $A(i \to \varphi)$ and $E(i \wedge \varphi)$ mean exactly the same thing as $@_i\varphi$. To some extent it's a matter of taste which approach is adopted, though it's worth knowing that adding nominals and satisfaction operators to the basic modal language does not take us out of PSPACE (see [4]), whereas adding the global modality (even if we don't add nominals) leads to an EXPTIME-complete satisfaction problem (see [39]).

Binders

A great deal could be said about binding. The basic idea is this: nominals, although they are formulas, are rather like the constants of first-order languages. So why not make it possible to *bind* occurrences of nominals, thereby increasing the expressive power still further? In fact, at least two ways of binding have been broadly investigated in the hybrid literature, namely *local binding* with the $\downarrow$-binder, and *global binding* with $\exists$ and $\forall$.

Roughly speaking, $\downarrow$ binds a nominal to the point of evaluation. Actually, just as first-order logic draws a distinction between constants and variables, hybrid logic draws a distinction between *state variables* and nominals. Syntactically, state variables are just like nominals, except that they can be bound and nominals can't. So it would be more accurate to say that $\downarrow$ binds a state variable to the point of evaluation. For example, consider the following formula:

$$\downarrow x \neg \Diamond x.$$

This names the current state x, and then insists that it is not possible to make an R-transition to the state named x. This, of course, will be true precisely when the state at which we are evaluating is not R-related to itself. To put it another way, it is a formula which distinguishes reflexive from irreflexive points in *any* model. No formula in ordinary modal logic, or even ordinary modal logic enriched with both nominals and satisfaction operators, can draw this distinction. The expressive power of $\downarrow$ is fully classified in [5].

Once you're accustomed to the idea of binding, it's tempting to go the whole hog and use the classical quantifiers $\exists$ and $\forall$, resulting in formulas like the following:

$$\exists x @_x \Diamond x.$$

This says: "somewhere in the the model there is a point x, and at the point named x it is possible to make an R-transition to the point named x," or more simply "the model contains a reflexive point."

A number of different binders are used in this special issue. Seligman uses the $\exists$ and $\forall$ binders as part of his proof theoretical investigation, Marx uses $\downarrow$ in the setting of relational algebra, and van Eijk *et al.* make use of a novel form of binding to describe network topologies.

That completes our bird's-eye view of hybrid logic. As we hope is now clear, hybrid logic offers something that ordinary modal logic does not — and yet the 'fit' between modal and hybrid logic is so organic that it would be artificial to regard them as separate disciplines. To give what may be a useful analogy: just as the classical logician can move freely between first-order languages with and without equality, we believe users of modal logic can (and should) add, discard (or invent!) hybrid apparatus as the need arises.

3 Historical and Bibliographical Remarks

The previous section told you what hybrid logics are, but not where they came from. In fact, hybrid logic has been around a surprisingly long time, and here we'll sketch their history and draw attention to a number of papers and other resources which may be useful to readers of this special issue.

Hybrid logic was introduced by Arthur Prior, the inventor of temporal logic, in the mid 1960s, and it played a fundemental role in his philosophical work. Drawing on McTaggart's [29] distinction between conceiving of time in terms of the A-series of past, present and future and the B-series of earlier and later, Prior introduced two logical systems. The T-calculus was intended to capture the A-series perspective, and used the tense operators F and P and variables ranging over propositions; it's what we would today call basic temporal (or tense) logic. The U-calculus aimed to capture the properties of the B-series, and made use of variables ranging over instants of time; it's essentially what a contemporary modal logician would call the temporal correspondence language.

Now, Prior viewed the A-series conception as fundamental, and wanted to show that 'tensed talk' could express everything that the U-calculus could. Unfortunately, the T-calculus (ordinary tense logic) is obviously weaker than the U-calculus (the temporal correspondence language) and Prior was well aware of this. Hybridization was Prior's response. In Chapter V.6 of [34], he enriches the T-calculus with instant-variables, allows them to be bound by $\forall$ and $\exists$, and adds (a variant of) the global modality. He called this "the third grade of tense-logical involvement" in [35, Chapter XI] and showed that the resulting system was strong enough to capture the U-calculus. In short, hybridization cleared the barrier to Prior's philosophical program of establishing the priority of tensed talk.

The technical development of hybrid logic was initiated by Prior's then student Robert Bull. In [17], a paper published in 1970, Bull proves a completeness result for a hybrid logic containing nominals, the $\forall$ and $\exists$ binders, the universal modality, and also *path nominals*. These 'name' branches in tree-like models of time by being true at all and only the points on the branch, thus they pick out a possible 'course of events.' Bull included path nominals in his hybrid logic — a decade before branching time logics were studied in theoretical computer science — because of Prior's interest in non-deterministic models of time. Bull demonstrates the relevance of the Henkin construction to hybrid logic, notes the ease with which richer logics can be dealt with, and suggests a novel approach (via non-standard models of set theory) to completeness theory.

The subject then seems to have lain dormant for over a decade until hybrid languages were independently reinvented by a group of logicians from Sofia, Bulgaria. George Gargov, Solomon Passy and Tinko Tinchev were interested in obtaining neat axiomatizations of various operations in Propositional Dynamic Logic. The problem here is that while certain operations (for example, union of programs) are easy to capture (union simply requires the axiom $\langle\alpha \cup \beta\rangle p \leftrightarrow \langle\alpha\rangle p \vee \langle\beta\rangle p$), a simple axiomatization of operations such as intersection or complement call for extra expressive power. In [30] it is shown that the addition of nominals is enough to provide succinct and natural characterization of intersection and complement. Moreover, the addition of other kind of "constants" to the language permits the representation of notions like determinism and looping [23] to be captured relatively straightforwardly. In addition, the work of the Sofia school showed how nominals could also be used to simplify the construction of models during completeness proof [31]. We strongly recommend [32] to readers of this special issue: it's a classic investigation of hybrid logic and the results and techniques remain relevant to contemporary concerns.

Hybrid logic entered its current phase in the 1990s, when a new generation of logicians (many of whom are represented in this special issue) became involved. The research of this period has lead in many new directions, but one general theme stands out: the exploration of weaker languages. For example, while Robert Bull and the Sofia school had realised that Henkin arguments could be used to prove completeness, their approaches require the use of relatively powerful hybrid languages. Similarly, in the 1990s it was realized that binding did not have to mean classical binding with $\forall$ and $\exists$, and the $\downarrow$ binder was isolated. For early papers in this phase, see [22, 11].

For the Paste rule, which permits Henkin methods to be used in the basic hybrid language, see [16]. For early work on $\downarrow$ see [24, 15], and for the current state of play see [5]. Hybrid proof theory, from a variety of perspectives, has blossomed in recent years, and we draw the readers attention to [38, 41, 18, 19, 12]. For interpolation results see [5], and for computational complexity, see [4].

That pretty much concludes the historical sketch — but it's worth stressing that we have only discussed what might be called 'mainstream' hybrid logic. One of the most exciting recent developments is the amount of work in neighbouring fields which echoes key hybrid logical themes. For example, the brand of labeled deduction developed by Basin, Matthews and Vigano [8, 9], links naturally with recent hybrid proof theory. Polish work on the logic of information systems and rough sets has lead to the evolution of what are essentially hybrid logics; see, for example, Konikowska [28]. For something close to hybrid logic, but developed from the perspective of first-order modal logic, see Gabbay and Malod [21]. Certain feature logics used in computational linguistics turn out to be hybrid logics (see [10, 36]) and in view of recent developments on HPSG, this line of work is overdue for a revival. Perhaps most interesting of all, however, is the increasing interplay between hybrid logic and description logic. For a detailed treatment of this link, see [3, 7]; for a hybrid-logical 'spypoint' argument being applied in description logic, see [40]; and for a recent description logic paper that makes a fundemental contribution to our understanding of hybrid logic, see [37].

If we have whetted your appetite, and you wish to learn more, then a good starting point is the Hybrid Logic Homepage

`http://www.hylo.net`.

Here you will find many of the papers mentioned above, and much other information besides. Also available there is the "Hybrid Logic Manifesto"[13] which is probably the most accessible introduction to the field currently available.

One final remark: we've mentioned a lot of theory, but what about implementation? As yet there is relatively little on offer, but this situation should soon change. On the Hybrid Logic Home page you can find a preliminary prototype of a prover for the basic hybrid language. This was implemented by Aljoscha Burchardt and Stephan Walter at Computerlinguistik, University of Saarland, as an undergraduate programming project (supervised by Patrick Blackburn), and you can experiment with the result over the web. At the University of Amsterdam, Carlos Areces and Juan Heguiabehere are implementing the direct resolution method presented in this special issue. Finally, the description logic community looks set to offer some useful tools, as Patel-Schneider recently announced that the next version of DLP [33] will support full nominals.

4 The Hybrid Logics Workshops

This special issue on hybrid logics was born from the HyLo 2000 Workshop, held in Birmingham as part of the Twelfth European Summer School in Logic, Language and Information (ESSLLI).

A year previously, the first HyLo workshop, HyLo'99, was organized by Patrick Blackburn at Computerlinguistik, University of Saarland, Germany. This closed workshop had two aims: to bring together researchers in hybrid languages to present recent developments, and to discuss how best to stimulate interest in the subject. HyLo'99 was the first workshop solely devoted to hybrid logic, and it made it clear that hybrid logics were gaining a new lease of life. At that time hybrid logics were starting to build strong links with different fields, notably description logic and labeled deduction, and its growing maturity meant that it could start to work as a theoretical framework for them.

After the first HyLo we knew we wanted to export hybrid logics to a wider audience, as we believed there were interesting ideas that many could profit from. ESSLLI was the perfect framework for these plans, with its wide mixture of backgrounds, covering logic, linguistics and computer science, and its broad attendance ranging from Masters and PhD students to leading researchers in these fields.

HyLo 2000: Bringing them All Together. HyLo at ESSLLI was always going to be quite different from the HyLo'99. For a start, it was a five day event, instead of a one day meeting. In addition, we didn't want another gathering of specialists in the area: we wanted to draw in as many people, from as varied backgrounds as possible.

The result was a complex formula, but we believe a successful one. HyLo 2000 was a mixture of workshop discussion, technical expositions, and tutorial presentation. We built the program around the invited speakers, and tried to fill in the details needed to draw a complete picture of the field.

To our delight, over 50 people attended HyLo 2000, a number we hadn't anticipated. We were particularly pleased, when on the second to last day, Martin Prior, Arthur Prior's son, was able to attend. The workshop seems to have filled its aim of raising the profile of hybrid logic: the number of visits to the hybrid logic web site increased dramatically following HyLo 2000 (we recently reached the 2300 hits). And, last but not least, HyLo 2000 provided the opportunity for this special issue.

5 Hybrid Logic in this Special Issue

After HyLo 2000, an open call for papers was circulated. From the papers received, the following five were accepted for publication:

Jerry Seligman. How to Create a Hybrid Calculus from its Semantic Theory. In this paper, Jerry Seligman takes us on an interesting journey. The satisfaction definition of most modal operators is specified in terms of first-order conditions. Hence we can always obtain a complete calculus for the basic logic characterizing any collection of such operators by appealing to a calculus which is complete for the full first-order language. Seligman shows here that by making use of the expressiveness provided by the hybrid apparatus, we can, step by step, transform a first-order sequent calculus into an internalized sequent calculus specifically tailored for a particular hybrid fragment.

Maarten Marx. Relation Algebra with Binders. Maarten Marx proposes extending the classical language of relation algebras with variables denoting individual elements in the domain and the hybrid binder $\downarrow$. This extension boosts the expressive power of the language to full elementary expressivity: any first-order property of binary relations can be now characterized. The most important part of Marx's article is the examples he discusses. These provide new perspectives on both relation algebra and hybrid logic.

Rogier van Eijk, Frank de Boer, Wiebe van der Hoek and John-Jules Meyer. Modal Logic with Bounded Quantification over Worlds. This paper develops a rather different kind of hybrid logics, from a rather different perspective. Driven by application issues (namely, to find the proper language to describe network topologies), van Eijk *et al.* arrive at a system which they describe as follows: "In comparison with standard hybrid languages, the logic covers separate mechanisms for navigation and for variable-binding and formalizes reasoning about the worlds of a model in terms of equational reasoning."

Carlos Areces, Hans de Nivelle and Maarten de Rijke. Direct Modal, Description and Hybrid Resolution. One of the reasons for hybridizing a modal logic is to try to improve its computational behavior. For example, as is discussed in [12], hybridization of basic modal logics leads to internalized labeled deduction. In this article, Areces *et al.* show how the same hybridization technique leads to simple resolution algorithms for modal and description logics. Going in the other direction, the use of resolution as a decision method for hybrid logics (which requires the use of paramodulation) sheds light on the view which regards hybrid logics as classical modal logics plus modal theories of state equality and state succession.

Valentin Goranko and Dimiter Vakarelov. Sahlqvist Formulas in Hybrid Polyadic Modal Logic. Goranko and Vakarelov investigate Sahlqvist's Theorem in the framework of hybrid logics. Building on the approach first discussed by the authors in [26], they provide a general description of hybrid formulas characterizing first-order properties of frames. A particularly interesting case is that of *reversive* languages, closed under all 'inverses' of modalities, because the minimal valuations arising in the computation of the first-order equivalents of Sahlqvist formulas are definable in such languages. This makes the proof of first-order definability and canonicity of these formulas a relatively simple syntactic exercise.

6 Other New Directions in Hybrid Logic

The articles in this special issue provide a reasonably broad perspective on hybrid logic, but they don't cover everything. Indeed, a sign of the field's health is that it is becoming increasingly difficult to keep abreast of developments — a novel situation in what has historically been a small field.

New developments in hybrid logic often come about by seeing the links which bind hybrid logic with other fields (this is certainly the case with the work relating hybrid logic to description logic, feature logic, and labeled deduction). And the same message keeps coming though: viewing other fields with hybrid eyes can lead to novel insights. Reciprocally, from these interactions hybrid logic acquires new proof methods, new directions for further development, and interesting problems to solve. We close this editorial by mentioning three new lines of work which will provide mind food for willing logic engineers.

Hybridizing First-Order Modal Logic. As we mentioned above, very expressive hybrid logics add different kind of binding and quantification across points to the underlying modal logic. But in most previous work, the underlying modal logic has been *propositional*. What happens when *first-order* modal logic is hybridized instead?

For over two decades, first-order modal logic has been something of a modal backwater. It is technically difficult terrain: coming up with general axiomatizations is hard, the area is plagued with frame incompleteness, and Craig interpolation and Beth definability are known to fail in a wide range of circumstances.

It is becoming clear that hybridizing first-order modal logic can cure many of these ills. For example, in a recent paper, Areces, Blackburn and Marx [6], show that equipping first-order modal logic with $\downarrow$ and @ yields systems with the Craig interpolation property (and hence Beth definability too). This holds for the logic of any class of frames definable in the bounded fragment of first-order logic, irrespective of whether constant, expanding, contracting, or arbitrary domains are assumed.

Why does hybridization open the door to general results in first-order modal logic? As in the propositional case, in essence because hybridization provides a frame language in which modal theories of equality and state succession can be formulated, and this makes it possible to blend first-order Henkin techniques with the use of modal canonical models.

Hybrid Logics and the μ-Calculus. A recent paper by Sattler and Vardi [37] investigates the logical language obtained by adding nominals and the global modality to the modal μ-calculus (with converse operators). Sattler and Vardi establish an ExpTime upper bound on the complexity of the satisfiability problem, thereby demonstrating the existence (as they put it) of a new ExpTime "Queen" logic.

The point is this. The use of the μ-binder over a modal logic with converse is already a powerful ExpTime complete system. But viewed from the perspective of (say) description logic, two familiar expressive weaknesses are apparent: general claims can't be formulated and individuals cannot be named. Adding the global modality and nominals solves these weaknesses, hence (from a description logic perspective) their result shows that it is possible to fully blend T-Box and A-Box reasoning in a system that can draw on the full resources of the modal μ-calculus with converse, without leaving ExpTime. Their system is a true "Queen," in which a number of important description logics can be straightforwardly embedded.

Just as interesting as the result, however, is the proof. Ordinary modal logics have the *tree property*: that is, satisfiable formulas can be satisfied on tree-based models, as a simple unraveling argument shows. But hybrid logics don't have this property: unraveling can destroy the requirement that nominals name unique points of the model. In spite of this, hybrid logics (without binders) are robustly decidable, and Sattler and Vardi's proof, which makes use of tree automata techniques, goes a long way towards explaining why.

Hybrid Quantification in Real Time Logics. Stéphane Demri and Valentin Goranko have recently called our attention to an interesting connection between the hybrid binder ↓, and certain operators introduced in the real time logics of Alur and Henzinger [2, 27]. In their proposal for a temporal logic modeling real time state transition systems, Alur and Henzinger were lead to models where each state has an associated value (which can be thought of as their time of execution). They then argue that a "retrieval operator" $x.\varphi$ is enough to express most interesting properties of such systems. For example,

$$\Box x.(p \rightarrow \Diamond y.(q \wedge y \leq x + 5)),$$

expresses that it is always the case that each request p is eventually followed by a response q within 5 units of time. Notice that x is similar to ↓: it creates on the fly, a syntactic reference to some "actual" value. There is a difference: whereas ↓ creates a transitory name for the actual state of evaluation, $x.$ retrieves the actual value associated with the state. But the ideas are closely related, and it seems likely that results can be transferred between the two lines of work.

This work also connects with first-order hybrid logic because, as the example above shows, the systems of Alur and Henzinger have a predicate structure which let us create terms like $x + 5$ and atomic formulas like $y \leq x + 5$. So $x.$ could be seen as a hybrid binder working on the first-order domain of each point, instead than on the domain of points.

As these examples show, neither HyLo 2000 nor this special issue managed to bring them *all* together — but that's simply because there were too many of them, surely an excellent sign. Modal logic is finding its way more and more each day into other fields (hardware and software verification, computational linguistics, spatial reasoning, knowledge representation, . . .), and each step leads to interesting new territory. Hybridization has an important role to play in this process, for it provides tools that can be justified on both theoretical and logic engineering grounds. Go ahead, have a look, and let us know what you think.

Acknowledgements

First of all we want to thank our anonymous reviewers for their helpful and often very detailed comments. Second we'd like to thank everyone who by their presentation or contribution to the panel discussion helped make HyLo 2000 such a success: Dov Gabbay, Valentin Goranko, Alexander Koller, Geert-Jan Kruijff, Guillaume Malod, Maarten Marx, Drew Moshier, Luciano Serafini, and Bill Wadge. Finally, we'd like to thank Dov Gabbay for letting us put this special issue together.

References

[1] J. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984.

[2] R. Alur and T. Henzinger. A really temporal logic. In *30th Annual Symposium on Foundations of Computer Science*, pages 164–169, New York, 1989. IEEE Computer Society Press.

[3] C. Areces. *Logic Engineering: The Case of Hybrid Logic and Description Logic.* PhD thesis, ILLC, University of Amsterdam, 2000.

[4] C. Areces, P. Blackburn, and M. Marx. The computational complexity of hybrid temporal logics. *Logic Journal of the IGPL*, 8(5):653–679, 2000.

[5] C. Areces, P. Blackburn, and M. Marx. Hybrid logics: Characterization, interpolation and complexity. *Journal of Symbolic Logic*, 2000. To appear.

[6] C. Areces, P. Blackburn, and M. Marx. Repairing the interpolation theorem in quantified modal logic. Submitted. Available at `http://www.hylo.net`, 2001.

[7] C. Areces and M. de Rijke. From description to hybrid logics, and back. In F. Wolter, H. Wansing, M. de Rijke, and M. Zakharyaschev, editors, *Advances in Modal Logic*, volume 3. CSLI Publications, 2001.

[8] D. Basin, S. Matthews, and L. Viganò. Labelled propositional modal logics: theory and practice. *Journal of Logic and Computation*, 7(6):685–717, 1997.

[9] D. Basin, S. Matthews, and L. Viganò. Natural deduction for non-classical logics. *Studia Logica*, 60(1):119–160, 1998.

[10] P. Blackburn. Modal logic and attribute value structures. In M. de Rijke, editor, *Diamonds and Defaults*, pages 19–65. Kluwer Academic Publishers, Dordrecht, 1993.

[11] P. Blackburn. Nominal tense logic. *Notre Dame Journal of Formal Logic*, 34(1):56–83, 1993.

[12] P. Blackburn. Internalizing labelled deduction. *Journal of Logic and Computation*, 10(1):137–168, 2000.

[13] P. Blackburn. Representation, reasoning, and relational structures: a Hybrid Logic manifesto. In C. Areces, E. Franconi, R. Goré, M. de Rijke, and H. Schlingloff, editors, *Methods for Modalities 1*, volume 8(3), pages 339–365. Logic Journal of the IGPL, 2000.

[14] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic.* Number 53 in Cabridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[15] P. Blackburn and J. Seligman. Hybrid languages. *Journal of Logic, Language and Information*, 4(3):251–272, 1995. Special issue on decompositions of first-order logic.

[16] P. Blackburn and M. Tzakova. Hybrid languages and temporal logic. *Logic Journal of the IGPL*, 7(1):27–54, 1999.

[17] R. Bull. An approach to tense logic. *Theoria*, 36:282–300, 1970.

[18] S. Demri. Sequent calculi for nominal tense logics: a step towards mechanization? In N. Murray, editor, *Conference on Tableaux Calculi and Related Methods (TABLEAUX), Saratoga Springs, USA*, volume 1617 of *LNAI*, pages 140–154. Springer Verlag, 1999.

[19] S. Demri and R. Goré. Cut-free display calculi for nominal tense logics. In N. Murray, editor, *Conference on Tableaux Calculi and Related Methods (TABLEAUX), Saratoga Springs, USA*, volume 1617 of *LNAI*, pages 155–170. Springer Verlag, 1999.

[20] D. Gabbay. *Labelled Deductive Systems. Vol. 1.* The Clarendon Press Oxford University Press, New York, 1996. Oxford Science Publications.

[21] D. Gabbay and G. Malod. Naming worlds in modal and temporal logic. *Journal of Logic, Language and Information*, 2000. To appear.

[22] G. Gargov and V. Goranko. Modal logic with names. *Journal of Philosophical Logic*, 22(6):607–636, 1993.

[23] G. Gargov and S. Passy. Determinism and looping in combinatory PDL. *Theoretical Computer Science*, 61(2-3):259–277, 1988.

[24] V. Goranko. Hierarchies of modal and temporal logics with reference pointers. *Journal of Logic, Language and Information*, 5(1):1–24, 1996.

[25] V. Goranko and S. Passy. Using the universal modality: gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992.

[26] V. Goranko and D. Vakarelov. Sahlqvist formulas unleashed in polyadic modal languages. In F. Wolter, H. Wansing, M. de Rijke, and M. Zakcharyaschev, editors, *Advances in Modal Logic, vol. 3*, Stanford, 2001. to appear in: CSLI Publications.

[27] T. Henzinger. Half-order modal logic: How to prove real-time properties. In *Proceedings of the 9th Annual Symposium on Principles of Distributed Computing*, pages 281–296. ACM Press, 1990.

[28] B. Konikowska. A logic for reasoning about relative similarity. *Studia Logica*, 58(1):185–226, 1997.

[29] J. McTaggart. The unreality of time. *Mind*, pages 457–474, 1908.

[30] S. Passy and T. Tinchev. PDL with data constants. *Information Processing Letters*, 20(1):35–41, 1985.

[31] S. Passy and T. Tinchev. Quantifiers in combinatory PDL: completeness, definability, incompleteness. In *Fundamentals of Computation Theory FCT 85*, volume 199 of *LNCS*, pages 512–519. Springer, 1985.

[32] S. Passy and T. Tinchev. An essay in combinatory dynamic logic. *Information and Computation*, 93(2):263–332, 1991.

[33] P. Patel-Schneider. DLP system description. In E. Franconi, G. De Giacomo, R. MacGregor, W. Nutt, and C. Welty, editors, *Proceedings of the 1998 International Workshop on Description Logics (DL'98)*, pages 87–89, 1998. DLP is available at `http://www.bell-labs.com/user/pfps`.

[34] A. Prior. *Past, Present and Future.* Oxford University Press, 1967.

[35] A. Prior. *Papers on Time and Tense.* University of Oxford Press, 1977.

[36] M. Reape. A feature value logic. In C. Rupp, M. Rosner, and R. Johnson, editors, *Constraints, Language and Computation*, Synthese Language Library, pages 77–110. Academic Press, 1994.

[37] U. Sattler and M. Vardi. The hybrid μ-calculus. In *Proceedings of IJCAR'01*, Siena, June 2001.

[38] J. Seligman. The logic of correct description. In M. de Rijke, editor, *Advances in Intensional Logic*, number 7 in Applied Logic Series, pages 107–135. Kluwer Academic Publishers, 1997.

[39] E. Spaan. *Complexity of Modal Logics.* PhD thesis, ILLC, University of Amsterdam, 1993.

[40] S. Tobies. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *Journal of Artificial Intelligence Research*, 12:199–217, May 2000.

[41] M. Tzakova. Tableaux calculi for hybrid logics. In N. Murray, editor, *Proceedings of the Conference on Tableaux Calculi and Related Methods (TABLEAUX), Saratoga Springs, USA*, volume 1617 of *LNAI*, pages 278–292. Springer Verlag, 1999.

Internalization: The Case of Hybrid Logics

Jeremy Seligman

Department of Philosophy, The University of Auckland
Private Bag 92019, Auckland, New Zealand
Email: j.seligman@auckland.ac.nz

Abstract

A sequent calculus for hybrid logics is developed from a calculus for classical predicat logic by a series of transformations. We formalize the semantic theory of hybrid logic using a sequent calculus for predicate logic plus axioms. This works, but it is ugly. The unnattractive features are removed one-by-one, until the final vestiges of the metalanguage can be set aside to reveal a fully internalized calculus. The techniques are quite general and can be applied to a wide range of hybrid and modal logics.

Keywords: Proof theory, cut-elimination, internalization, hybrid logic, modal loigc, subformula property, sequent calculus.

1 Introduction

The classical beauty of Gentzen's sequent calculus is obvious from first acquaintance. Each logical operator is precisely characterized by a pair of rules with perfect economy. The operators appear only in the conclusion of the rules, which are constructed from a tidy arrangement of subformulas. Every symbol occurs only in the place that best explains its function; nothing is wasted.

The fit between the geometry of sequent proofs and classical predicate logic is almost too perfect. When we try to use similar techniques with other logics, it never quite works. With intuitionistic logic the left-right symmetry is broken; with modal logics, there are ugly restrictions on the contexts. This paper is a contribution to our understanding of why this happens.

We propose a means of analyzing the form of a logic by the process of internalization. A calculus is *fully internalized* if the only symbols that occur in the rules of the calculus are symbols of the object language. No labels or special positional operators are allowed—only the geometry of syntactic structure and rules governing how logical symbols may be added and subtracted. Gentzen's sequent calculi are all fully internalized, as are many of the calculi proposed for modern applied logics. Yet almost all of them require the symmetry-breaking contortions of intuitionistic and modal logic.

Contrast the situation in proof theory with that in model theory. There, applied logics have a natural home in the world of Kripke structures. A semantic theory of relations can be constructed for almost all known logics, and each one is more-or-less as good as the others. The theory proposes a network of extra-logical machinery—accessibility relations and the like—in terms of which the logical operators are translated. This is an *external* approach to logic because it uses much beyond the syntax of the object language.

The passage from external semantic theory to fully internalized calculus is well-understood, especially among those logicians who can knock up a sequent calculus for a new modal logic before breakfast. The strategy involves an implicit translation of everything back into first-order predicate logic. The axioms of the semantic theory are typically first-order, and so their logical properties can be seen through their representation in the sequent calculus for classical first-order predicate logic. The trick is to see how to get the effect of the classical sequent rules using the syntax of the new logic.

In this paper, we formalize the process of internalization explicitly using, first, an expression of the external semantic theory in classical predicate logic, and then a series of transformations taking us to more-and-more internalized calculi, dropping the metalogical props one by one.

To illustrate the process, we focus on hybrid logics. Hybrid logics lie at the boundary between predicate logic and propositional modal logic, making them an especially appropriate focus for proof-theoretic techniques that also cross between these realms. Recent papers [3, 16, 8, 7] have developed a number of proof systems for hybrid logics, some of which are fully internalized. And, of course, they all have clearly formulated Kripke-style semantic theories. They form a perfect case study for a theory of internalization![1]

We begin Section 2 by reviewing the semantic theory of hybrid logics, which we express as a set of axioms in a formal metalanguage. In Section 3 we introduce a variant of the classical sequent calculus for predicate logic that is especially good at handling equality, and add axioms to capture the formal semantics of hybrid logics. Section 4 regains the Subformula Property, which was lost with the addition of axioms, by generating rules for the hybrid operators. The Subformula Property is essential to the process because it allows us to throw away extraneous rules while maintaining completeness. The resulting system still uses metalogical labels to control the flow of information in a proof, but these are removed with the use of the hybrid operators in Section 5.

2 Hybrid Languages and their Formal Semantics

Hybrid logics have a long history and a number of interesting applications. We refer the reader to [10] for a wealth of information about these logics and an extensive bibliography. Our present interest in hybrid logics is merely that of providing an example of a broad class of logics for which the internalization strategy is particularly appropriate.

[1]Naturally, the fit is too good to be a matter of chance. My awareness of the process of internalization came from earlier attempts at providing a sequent calculus for hybrid logics. These are documented in [3].

There are two routes to understanding hybrid logics. The first, and most common, is to think of them as propositional modal logics enriched with various devices from predicate logic to increase their expressive power. An alternative perspective, adopted here, sees them as the contextualization of predicate logic.

Let L be the language of predicate logic with individual variables x $(= x_1, x_2, \ldots)$, property symbols p $(= p_1, p_2, \ldots)$, and (binary) relation symbols r $(= r_1, r_2, \ldots)$. Closed formulas of L express properties of relational structures via the definition of truth in a structure. For example, the formula $\forall x\, rxx$ is satisfied by those structures having a reflexive r-relation.

Hybrid logic results from the contextualization of these properties to elements of a structure. We aim to express the contextual properties of an element of the structure using formulas with an implicit parameter that refers to that element. Any formula of L with one free variable expresses such a property—we move to hybrid logic by erasing the free variable and assuming that its reference is supplied as an implicit parameter.

A property symbol p combines with a variable x to give a formula px. Hide the x and you get the hybrid formula p, which can be combined with other formulas using the standard Boolean operators. The formula $\exists y\,(rxy \,\&\, py)$ becomes a modal formula $\Diamond_r p$ when the variable x is hidden and assumed to refer to the point of evaluation. All this is present in ordinary modal logic. We get the specifically hybrid operators by applying the same idea to a wider class of formulas. For example, when the x of $x = y$ is hidden we get the nominal y, but now as a formula that can be combined with other formulas using logical operators. In a context in which y refers to an element a of a structure, the new hybrid formulas y expresses the property of being identical to a.

The language H of hybrid logic is defined as follows. The atomic formulas of H are the individual variables and the property symbols. Complex formulas are built up using the Boolean operators $\vee$ and $\sim$ together with unary modal operators $\Diamond$,$\Diamond_r$, and $\exists x$. Negation duals of the operators, $\&$, $\Box$, $\Box_r$, and $\forall x$ are defined as abbreviations. The distinction between free and bound variables applies to H just as it does to L. The semantics of H is a straightforward adaption of the semantics of L. An interpretation for H is an interpretation for L, namely a relational structure $\mathbf{A}$ of type $(p_1, p_2, \ldots, r_1, r_2, \ldots)$ with a distinguished point, a in the domain $|\mathbf{A}|$ of $\mathbf{A}$[2]. Given an assignment g of elements of $|\mathbf{A}|$ to the variables, we define the relation $\vDash$ (*satisfies*) as follows:

$$\begin{array}{ll}
\mathbf{A}, a, g \vDash x & \text{if } g(x) = a \\
\mathbf{A}, a, g \vDash p & \text{if } a \in p^{\mathbf{A}} \\
\mathbf{A}, a, g \vDash {\sim}\varphi & \text{if not } \mathbf{A}, a, g \vDash \varphi \\
\mathbf{A}, a, g \vDash (\varphi \vee \psi) & \text{if either } \mathbf{A}, a, g \vDash \varphi \text{ or } \mathbf{A}, a, g \vDash \psi \\
\mathbf{A}, a, g \vDash \Diamond\varphi & \text{if for some } a' \in |\mathbf{A}|,\ \mathbf{A}, a', g \vDash \varphi \\
\mathbf{A}, a, g \vDash \Diamond_r\varphi & \text{if for some } a' \in |\mathbf{A}|,\ \langle a, a' \rangle \in r^{\mathbf{A}} \text{ and } \mathbf{A}, a', g \vDash \varphi \\
\mathbf{A}, a, g \vDash \exists x\, \varphi & \text{if for some } a' \in |\mathbf{A}|,\ \mathbf{A}, a, g[{}^{x}_{a'}] \vDash \varphi
\end{array}$$

where, as usual, $g[{}^{x}_{a'}](y){=}a'$ if $y = x$, and $g(y)$ otherwise.

[2]A *relational structure* of type $(p_1, p_2, \ldots, r_1, r_2, \ldots)$ is a set $|\mathbf{A}|$ together with a subset $p_i^{\mathbf{A}}$ of $|\mathbf{A}|$ for each p_i and a subset $r_i^{\mathbf{A}}$ of $|\mathbf{A}|^2$ for each r_i.

This is immediately recognizable as containing the Kripke semantics for a language of modal logic in which p is a propositional variable and $\Diamond$ is the S5 modal operator. $\Diamond_r$ is also a normal modal operator, interpreted using the accessibility relation $r^{\mathbf{A}}$. What has been added is distinctly hybrid: individual variables occurring as formulas (*nominals*) and the modal quantifier $\exists x$

Sequents of the languages mentioned above are expressions of the form $\Gamma \longrightarrow \Delta$, in which Γ and Δ are lists of formulas. Such a sequent is *valid* if every $\mathbf{A}$, a,g satisfying every formula in Γ also satisfies some formula in Δ.

The hybrid character of the language is further developed with the operators $@_x$ and $\downarrow_x$. The only syntactic difference between these is that the *downarrow* operator $\downarrow_x$ binds its variable, whereas the *at* operator, $@_x$, does not. They are interpreted in a relational structure as follows:

$$\begin{array}{ll} \mathbf{A}, a, g \models @_x\varphi & \text{if } \mathbf{A}, g(x), g \models \varphi \\ \mathbf{A}, a, g \models \downarrow_x \varphi & \text{if } \mathbf{A}, a, g[^x_a] \models \varphi \end{array}$$

Clearly, there are many other possibilities for hybrid operators, and yet there is already a kind of expressive completeness. The above two operators are definable in H as $\Diamond(x \,\&\, \varphi)$ and $\exists x\,(x \,\&\, \varphi)$. In fact, every operator definable in the predicate language L is also definable in H (see [4]).

It may seem a little strange to focus on such a richly expressive language as H. Much of the research on hybrid languages (for example, [5, 6, 2, 1]) has concentrated on fragments of H with much less expressive power and correspondingly lower computational complexity. We ensure that the rules developed for H can be applied to fragments by insisting that they have a subformula property. On the strictest interpretation, a rule is said to have the Subformula Property if every formula occurring in an application of a rule is a subformula of the conclusion. A complete calculus in which no rule involves more than one operator and every rule has the Subformula Property is guaranteed to be *modular*: select any set of operators and the set of rules using those operators will be complete for the fragment of the language formed from those operators. Similar results can be obtained even when the Subformula Property is weakened to allow a restricted class of formulas for each rule, so long as the fragment incudes the formulas in the restricted classes.

We are now ready to formalize the semantic theory of H. Let M (for 'meta') be the language of predicate logic extending L with new atomic formulas of the form $x\colon\varphi$, where φ is a formula of H. The free variables of $x\colon\varphi$ are x together with the free variables of φ. To make it (conceptually) clear that M is a language of predicate logic, we should have introduced a fresh $n+1$-ary relation symbol for each formula φ of H with n free variables. But this would have made the formulas much less clear (to read) and so we allow what is obviously a merely notational variant.

The semantics of H can be expressed directly in M as follows:

$$\begin{array}{ll} \forall z\,\forall x\,(z\colon x \equiv x = z) & \forall z\,(z\colon \Diamond\varphi \equiv \exists x\, x\colon \varphi) \\ \forall z\,(z\colon p \equiv pz) & \forall z\,(z\colon \Diamond_r\varphi \equiv \exists x\,(rzx \,\&\, x\colon \varphi)) \\ \forall z\,(z\colon {\sim}\varphi \equiv {\sim} z\colon \varphi) & \forall z\,\forall x\,(z\colon @_x\varphi \equiv x\colon \varphi) \\ \forall z\,(z\colon (\varphi \vee \psi) \equiv (z\colon \varphi \vee z\colon \psi)) & \forall z\,(z\colon \downarrow_x \varphi \equiv z\colon \varphi[^x_z]) \\ \forall z\,(z\colon \exists x\,\varphi \equiv \exists x\, z\colon \varphi) & \end{array}$$

Similar sentences could be produced for any other first-order definable hybrid operator. Let Θ be the above set of sentences of M. A sequent $\Gamma \longrightarrow \Delta$ is Θ-*valid* if for every model $\mathbf{A}$ of Θ and every assignment g, if $\mathbf{A}, g$ satisfies all the formulas in Γ, then it also satisfies some of the formulas in Δ. From such a characterization of the semantics of H, we get the following lemma. For each list Γ of formulas of H, let $u{:}\,\Gamma$ be the list of formulas $u{:}\,\varphi$, for each φ in Γ.

Lemma 1 A sequent $\Gamma \longrightarrow \Delta$ of H is valid if and only if $u{:}\,\Gamma \longrightarrow u{:}\,\Delta$ is Θ-valid.

Proof It is enough to observe that there is a one-one correspondence between models $\mathbf{B}$ of Θ and relational structures $\mathbf{A}$ for H, such that $|\mathbf{A}|{=}|\mathbf{B}|$ and for every assignment g, element a of the domain, formula φ of H, and variable x not in φ,

$$\mathbf{A}, a, g \models \varphi \text{ if and only if } \mathbf{B}, g[^x_a] \models x{:}\,\varphi$$

For any Θ-model $\mathbf{B}$, the corresponding structure $\mathbf{A}$ is just the reduct to properties $p_1, p_2, \ldots$ and relations $r_1, r_2, \ldots$. The correspondence is one-one because the theory Θ fixes the interpretation of all of the new relation symbols.

3 Sequent Calculus for Θ-validity

Valid sequents in M can be generated in a uniform way using the sequent calculus SM shown in Figure 1. Here φ and ψ range over formulas of M and Γ and Δ range over lists of formulas of M. We write $\Gamma \approx \Delta$ to mean that Γ and Δ contain the same *set* of formulas. For any expression σ, we write $\sigma[^x_y]$ for the result of replacing all free occurrences of x in σ by y.

The rules are written in *horizontal notation*, with premises listed to the left of '$\Rightarrow$' and conclusion to the right[3].

For proof-theoretic purposes it is useful to make a global decision about which variables can occur free in a sequent. Let $u_1, u_2, \ldots$ be a distinguished class of individual variables, called *parameters*. We restrict our attention to sequents whose free variables are all parameters and whose bound variables are not parameters, so that awkward clashes of variables can be avoided. We use u, v, w to range over parameters and x, y, z to range over the more inclusive class of individual variables.
The sequent to the right of the arrow $\Rightarrow$ is the *conclusion* of the rule; those on the left are its *premises*. A sequent $\Gamma \longrightarrow \Delta$ is a *theorem* of a set of rules, such as SM, if it is generated by them. In other words, a sequent is a theorem if it is the conclusion of a rule whose premises (if it has premises) are also theorems. The generation tree for a theorem is called a *proof*[4]. The calculus SM has the celebrated *Subformula Property*: every formula occurring in a proof of $\Gamma \longrightarrow \Delta$ is a subformula of a formula in Γ or Δ. The rule of *Weakening* (W),

$$\Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma', \Gamma \longrightarrow \Delta, \Delta'$$

[3]This is equivalent to *vertical notation*, often seen in textbooks, in which premises and conclusion are separated by a deduction line of the kind used in proofs. I prefer the horizontal notation because it is less cumbersome and makes a clearer distinction between the *rules* of a system and the *proofs* produced when the rules are applied.

[4]Note that when we draw a proof as a tree, any application of S will not be shown unless it is especially significant.

Structural Rules	
I	$\Rightarrow \quad \varphi, \Gamma \longrightarrow \Delta, \varphi.$
S	$\Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma' \longrightarrow \Delta'$ if $\Gamma \approx \Gamma'$ and $\Delta \approx \Delta'$.
Logical Rules	
$\sim$L	$\Gamma \longrightarrow \Delta, \varphi \quad \Rightarrow \quad {\sim}\varphi, \Gamma \longrightarrow \Delta.$
$\sim$R	$\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, {\sim}\varphi.$
$\vee$L	$\varphi, \Gamma \longrightarrow \Delta;\ \psi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad (\varphi \vee \psi), \Gamma \longrightarrow \Delta$
$\vee$R	$\Gamma \longrightarrow \Delta, \varphi, \psi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, (\varphi \vee \psi)$
$\exists$L	$\varphi[\substack{x\\u}], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \exists x\, \varphi, \Gamma \longrightarrow \Delta$ if u does not occur in φ, Γ, Δ.
$\exists$R	$\Gamma \longrightarrow \Delta, \varphi[\substack{x\\u}] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, \exists x\, \varphi$
$=$L$_1$	$u = v, \Gamma[\substack{w\\u}] \longrightarrow \Delta[\substack{w\\u}] \quad \Rightarrow \quad u = v, \Gamma[\substack{w\\v}] \longrightarrow \Delta[\substack{w\\v}].$
$=$L$_2$	$u = v, \Gamma[\substack{w\\v}] \longrightarrow \Delta[\substack{w\\v}] \quad \Rightarrow \quad u = v, \Gamma[\substack{w\\u}] \longrightarrow \Delta[\substack{w\\u}].$
$=$R	$\Rightarrow \quad \Gamma \longrightarrow \Delta, u = u$

Figure 1: The sequent calculus SM

is *admissible*, which means that its addition to the set of rules will not allow us to prove more theorems.
Of the rules of SM, only the Barwise equality rules, $=$L$_1$ and $=$L$_2$, may be unfamiliar[5]. Together they allow the replacement of any number of occurrences of u by v and v by u in a sequent containing $u = v$ on the left side. An advantage of the Barwise rules is the straightforward way in which the following result may be stated:

Theorem 2 The rule of *Cut* (C)

$$\Gamma \longrightarrow \Delta, \varphi;\ \varphi, \Gamma' \longrightarrow \Delta' \quad \Rightarrow \quad \Gamma, \Gamma' \longrightarrow \Delta, \Delta'$$

is admissible in SM.

Proof The theorem is proved by the method of cut-elimination: we show that every application of C—called a *cut*—can be pushed up the proof tree until it falls off the leaves. Technically, this is done by assigning a number to each cut—its *cut rank*—and transforming the proof so as to reduce distance between the cuts of maximal rank and the leaves. Cuts at the leaves are shown to be replaceable by axioms (I or $=$R, in this case). The transformations are of two kinds. In the primary case, the cut formula is assumed to be the principal formula of both rules immediate above the cut. Typically, the transformation replaces the cut with one or more cuts of lower rank. For example, if the cut formula is ${\sim}\varphi$ we have

$$\dfrac{\dfrac{\overset{\pi_1}{\varphi, \Gamma \longrightarrow \Delta}}{\Gamma \longrightarrow \Delta, {\sim}\varphi}\,{\sim}\mathrm{R} \qquad \dfrac{\overset{\pi_2}{\Gamma' \longrightarrow \Delta', \varphi}}{{\sim}\varphi, \Gamma' \longrightarrow \Delta'}\,{\sim}\mathrm{L}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\mathrm{C} \quad \rightsquigarrow \quad \dfrac{\overset{\pi_2}{\Gamma' \longrightarrow \Delta', \varphi} \quad \overset{\pi_1}{\varphi, \Gamma \longrightarrow \Delta}}{\Gamma', \Gamma \longrightarrow \Delta', \Delta}\,\mathrm{C}$$

[5]They were used by Jon Barwise in some early work on infinitary logic.

The cut rank of φ is less than that of ${\sim}\varphi$, so this is an improvement. The second kind of transformation occurs when the cut formula is not the principal formula of the final rule of one of the two branches. In this case, we must show how the cut can be moved up that branch, closer to the leaf. For example,

$$\dfrac{\dfrac{\overset{\pi_1}{\varphi, \Gamma \longrightarrow \Delta, \exists x\,\psi}}{\Gamma \longrightarrow \Delta, {\sim}\varphi, \exists x\,\psi}\,{\sim}\mathrm{R} \qquad \overset{\pi_2}{\exists x\,\psi, \Gamma' \longrightarrow \Delta'}}{\Gamma, \Gamma' \longrightarrow \Delta, {\sim}\varphi, \Delta'}\,\mathrm{C} \quad\rightsquigarrow\quad \dfrac{\dfrac{\overset{\pi_1}{\varphi, \Gamma \longrightarrow \Delta, \exists x\,\psi} \quad \overset{\pi_2}{\exists x\,\psi, \Gamma' \longrightarrow \Delta'}}{\varphi, \Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\mathrm{C}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta', {\sim}\varphi}\,{\sim}\mathrm{R}$$

For the system without the equality rules, the transformations are all standard (see, for example, [15]). The treatment of the equality rules deserves some comment. In the primary case, there is no difficulty at all in removing cuts whose cut formula is an equation:

$$\dfrac{\dfrac{}{\Gamma \longrightarrow \Delta, u = u}\,{=}\mathrm{R} \qquad \dfrac{\overset{\pi}{u = u, \Gamma'\genfrac{[}{]}{0pt}{}{w}{u} \longrightarrow \Delta'\genfrac{[}{]}{0pt}{}{w}{u}}}{u = u, \Gamma'\genfrac{[}{]}{0pt}{}{w}{u} \longrightarrow \Delta'\genfrac{[}{]}{0pt}{}{w}{u}}\,{=}\mathrm{L}_1}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\mathrm{C}$$

$$\rightsquigarrow\quad \dfrac{\dfrac{}{\Gamma \longrightarrow \Delta, u = u}\,{=}\mathrm{R} \qquad \overset{\pi}{u = u, \Gamma'\genfrac{[}{]}{0pt}{}{w}{u} \longrightarrow \Delta'\genfrac{[}{]}{0pt}{}{w}{u}}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\mathrm{C}$$

But the secondary case is a little more involved:

$$\dfrac{\dfrac{\overset{\pi_1}{u = v, \Gamma\genfrac{[}{]}{0pt}{}{w}{u} \longrightarrow \Delta\genfrac{[}{]}{0pt}{}{w}{u}, \varphi\genfrac{[}{]}{0pt}{}{w}{u}}}{u = v, \Gamma\genfrac{[}{]}{0pt}{}{w}{v} \longrightarrow \Delta\genfrac{[}{]}{0pt}{}{w}{v}, \varphi\genfrac{[}{]}{0pt}{}{w}{v}}\,{=}\mathrm{L}_1 \qquad \overset{\pi_2}{\varphi\genfrac{[}{]}{0pt}{}{w}{v}, \Gamma' \longrightarrow \Delta'}}{u = v, \Gamma\genfrac{[}{]}{0pt}{}{w}{v}, \Gamma' \longrightarrow \Delta\genfrac{[}{]}{0pt}{}{w}{v}, \Delta'}\,\mathrm{C}$$

$$\rightsquigarrow\quad \dfrac{\dfrac{\dfrac{\overset{\pi_1^*\genfrac{[}{]}{0pt}{}{v}{u}}{u = u, \Gamma\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u} \longrightarrow \Delta\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u}, \varphi\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u}} \quad \overset{\pi_2^*\genfrac{[}{]}{0pt}{}{v}{u}}{\varphi\genfrac{[}{]}{0pt}{}{w}{v}\genfrac{[}{]}{0pt}{}{v}{u}, \Gamma'\genfrac{[}{]}{0pt}{}{v}{u} \longrightarrow \Delta'\genfrac{[}{]}{0pt}{}{v}{u}}}{u = u, \Gamma\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u}, \Gamma'\genfrac{[}{]}{0pt}{}{v}{u} \longrightarrow \Delta\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u}, \Delta'\genfrac{[}{]}{0pt}{}{v}{u}}\,\mathrm{C}}{u = v, u = u, \Gamma\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u}, \Gamma'\genfrac{[}{]}{0pt}{}{v}{u} \longrightarrow \Delta\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u}, \Delta'\genfrac{[}{]}{0pt}{}{v}{u}}\,\mathrm{W}}{u = v, u = v, \Gamma\genfrac{[}{]}{0pt}{}{w}{u}, \Gamma' \longrightarrow \Delta\genfrac{[}{]}{0pt}{}{w}{u}, \Delta'}\,{=}\mathrm{L}_2$$

The problem here is that the cut formula $\varphi\genfrac{[}{]}{0pt}{}{w}{v}$ becomes $\varphi\genfrac{[}{]}{0pt}{}{w}{u}$ on the left side, but stays as $\varphi\genfrac{[}{]}{0pt}{}{w}{v}$ on the right side. The solution is to replace v by u throughout the two branches, after first changing any parameters introduced by the restricted quantifier rule $\exists$L[6]. This does not increase the length of the branches. The cut can be moved up the left branch because $\varphi\genfrac{[}{]}{0pt}{}{w}{u}\genfrac{[}{]}{0pt}{}{v}{u}$ is identical to $\varphi\genfrac{[}{]}{0pt}{}{w}{v}\genfrac{[}{]}{0pt}{}{v}{u}$. After the new cut, the vs can be put back in their proper places using $=\mathrm{L}_2$[7].

[6]We write π_i^* for the result of renaming the parameters in π_i that are introduced by a restricted rule, such as $\exists$L.

[7]In the manuscript, [14], cut-elimination is proved in a less direct manner, using redundant equality axioms in addition to the rules. In effect, the substitution we perform above is there postponed until the elimination algorithm reaches the leaves of the tree.

Theorem 3 A sequent of M is valid if and only if it is a theorem of SM.

Proof To demonstrate soundness (the 'if' direction), we need only observe that if the premises of an SM rule are valid, then so is the conclusion. Completeness (the 'only if' direction) follows from Theorem 2 and the completeness of the predicate calculus with identity, given the derivability of standard sequent axioms of predicate logic[8]. Since the only non-standard rules are the equality rules, we need only observe that reflexivity, symmetry, and transitivity are easily derived:

$$\dfrac{}{\longrightarrow u = u}\;{=}\mathrm{R} \qquad \dfrac{\dfrac{}{u = v \longrightarrow u = u}\;{=}\mathrm{R}}{u = v \longrightarrow v = u}\;{=}\mathrm{L}_2 \qquad \dfrac{\dfrac{\dfrac{}{u = v, v = w \longrightarrow u = u}\;{=}\mathrm{R}}{u = v, v = w \longrightarrow u = v}\;{=}\mathrm{L}_2}{u = v, v = w \longrightarrow u = w}\;{=}\mathrm{L}_2$$

The Subformula Property allows us to extend this result to all fragments of M obtained by removing logical operators.

Corollary 4 For any fragment M' of M that is closed under subformulas—i.e., all subformulas of formulas in M' are also in M'—let SM' be the set of rules of SM that involve operators occurring in M', together with the structural rules. Then a sequent of M' is valid if and only if it is a theorem of SM'.

Proof Soundness (the 'if' direction) follows from the soundness of SM. For the converse, suppose that $\Gamma \longrightarrow \Delta$ is a valid sequent of M'. Then, by Theorem 3 it is a theorem of SM. By the Subformula Property, its proof contains only formulas in M'. The rules used in the proof are therefore in SM', and so $\Gamma \longrightarrow \Delta$ is a theorem of SM'.

These results are easily strengthened to deal with Θ-validity.

Corollary 5 A sequent $\Gamma \longrightarrow \Delta$ of M is Θ-valid if and only if there are formulas $\varphi_1, \ldots, \varphi_n$ in Θ such that $\varphi_1, \ldots, \varphi_n, \Gamma \longrightarrow \Delta$ is a theorem of SM.

Proof By the compactness of predicate logic, and Theorem 3.

The simplest way of extending SM to a calculus for Θ-validity, is to add a new axiom $\Gamma \quad \Rightarrow \quad \Delta, \varphi$ for each sentence φ in Θ. Call the set of these new axioms AΘ.

Theorem 6 A sequent of M is a theorem of S(M+AΘ+C) if and only if it is Θ-valid.

Proof The new rules are obviously Θ-valid, and all the old rules preserve validity and hence also Θ-validity. So, every theorem of S(M+AΘ+C) is Θ-valid. Conversely, suppose that $\Gamma \longrightarrow \Delta$ is Θ-valid. By Corollary 5, there are formulas $\varphi_1, \ldots, \varphi_n$ in Θ, such that $\varphi_1, \ldots, \varphi_n, \Gamma \longrightarrow \Delta$ is a theorem of SM. But $\longrightarrow \varphi_i$ is in AΘ for each i, and so by n applications of the C rule, we can show that $\Gamma \longrightarrow \Delta$ is a theorem of S(M+AΘ+C).

Unfortunately, the C rules cannot be entirely eliminated from proofs in S(M+AΘ+C), which lacks the Subformula Property. This limits the theoretical utility of the calculus greatly.

Derived Logical Rules	
&L	$\varphi, \psi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad (\varphi \mathbin{\&} \psi), \Gamma \longrightarrow \Delta.$
&R	$\Gamma \longrightarrow \Delta, \varphi;\ \Gamma \longrightarrow \Delta, \psi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, (\varphi \mathbin{\&} \psi)$
⊃L	$\Gamma \longrightarrow \Delta, \varphi;\ \psi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad (\varphi \supset \psi), \Gamma \longrightarrow \Delta$
⊃R	$\varphi, \Gamma \longrightarrow \Delta, \psi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, (\varphi \supset \psi).$
≡L	$\varphi, \psi, \Gamma \longrightarrow \Delta;\ \Gamma \longrightarrow \Delta, \varphi, \psi \quad \Rightarrow \quad (\varphi \equiv \psi), \Gamma \longrightarrow \Delta$
≡R	$\varphi, \Gamma \longrightarrow \Delta, \psi;\ \psi, \Gamma \longrightarrow \Delta, \varphi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, (\varphi \equiv \psi)$
∀L	$\varphi[^x_u], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \forall x\, \varphi, \Gamma \longrightarrow \Delta$
∀R	$\Gamma \longrightarrow \Delta, \varphi[^x_u] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, \forall x\, \varphi$ if u does not occur in φ, Γ, Δ.

Figure 2: Derived rules of SM

Before addressing this problem, let us just note that with standard abbreviations, the rules listed in Figure 2 can all be derived in SM

4 Regaining the Subformula Property

The barrier to the elimination of C from S(M+AΘ+C), are cuts involving the formulas of Θ. For example, the following cut cannot be eliminated. Let π be the proof

$$\dfrac{\dfrac{\dfrac{}{u{:}\,\exists x\,\Diamond_r x, \exists x\,u{:}\,\Diamond_r x, v{:}\,\Diamond_r u, u{:}\,\Diamond_r v \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\mathrm{I} \qquad \dfrac{\dfrac{}{v{:}\,\Diamond_r u, u{:}\,\Diamond_r v \longrightarrow u{:}\,\exists x\,\Diamond_r x, u{:}\,\exists x\,\Diamond_r x, u{:}\,\Diamond_r v}\,\mathrm{I}}{v{:}\,\Diamond_r u, u{:}\,\Diamond_r v \longrightarrow u{:}\,\exists x\,\Diamond_r x, u{:}\,\exists x\,\Diamond_r x, \exists x\,u{:}\,\Diamond_r x}\,\exists\mathrm{R}}{(u{:}\,\exists x\,\Diamond_r x \equiv \exists x\,u{:}\,\Diamond_r x), v{:}\,\Diamond_r u, u{:}\,\Diamond_r v \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\equiv\mathrm{L}}{\forall z\,(z{:}\,\exists x\,\Diamond_r x \equiv \exists x\,z{:}\,\Diamond_r x), v{:}\,\Diamond_r u, u{:}\,\Diamond_r v \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\forall\mathrm{L}$$

then

$$\dfrac{\dfrac{}{\longrightarrow \forall z\,(z{:}\,\exists x\,\Diamond_r x \equiv \exists x\,z{:}\,\Diamond_r x)}\,\Theta \qquad \dfrac{\dfrac{\dfrac{\dfrac{\pi}{\forall z\,(z{:}\,\exists x\,\Diamond_r x \equiv \exists x\,z{:}\,\Diamond_r x), (v{:}\,\Diamond_r u \mathbin{\&} u{:}\,\Diamond_r v) \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\&\mathrm{L}}{\forall z\,(z{:}\,\exists x\,\Diamond_r x \equiv \exists x\,z{:}\,\Diamond_r x), \exists y\,(y{:}\,\Diamond_r u \mathbin{\&} u{:}\,\Diamond_r y) \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\exists\mathrm{L}}{\forall z\,(z{:}\,\exists x\,\Diamond_r x \equiv \exists x\,z{:}\,\Diamond_r x), \forall x\,\exists y\,(y{:}\,\Diamond_r x \mathbin{\&} x{:}\,\Diamond_r y) \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\forall\mathrm{L}}{\forall z\,(z{:}\,\exists x\,\Diamond_r x \equiv \exists x\,z{:}\,\Diamond_r x), \forall x\,\exists y\,(y{:}\,\Diamond_r x \mathbin{\&} x{:}\,\Diamond_r y) \longrightarrow \forall y\,y{:}\,\exists x\,\Diamond_r x}\,\forall\mathrm{R}}{\forall x\,\exists y\,(y{:}\,\Diamond_r x \mathbin{\&} x{:}\,\Diamond_r y) \longrightarrow \forall y\,y{:}\,\exists x\,\Diamond_r x}\,\mathrm{C}$$

[8]Theorem 2 is needed for logical closure. For example, we need to know that if $(\varphi \supset \psi), \varphi \longrightarrow \psi$ is an axiom (MP) and $\longrightarrow \varphi$ and $\longrightarrow (\varphi \supset \psi)$ are theorems then so is $\longrightarrow \psi$.

The cut can be pushed up the tree as far as the application of ∀L, but no further:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\longrightarrow \forall z\,(z{:}\,\exists x\,\Diamond_r x \equiv \exists x\, z{:}\,\Diamond_r x)}\,\Theta \qquad \pi}{v{:}\,\Diamond_r u,\, u{:}\,\Diamond_r v \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\text{C}}{(v{:}\,\Diamond_r u \;\&\; u{:}\,\Diamond_r v) \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\&\text{L}}{\exists y\,(y{:}\,\Diamond_r u \;\&\; u{:}\,\Diamond_r y) \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\exists\text{L}}{\forall x\,\exists y\,(y{:}\,\Diamond_r x \;\&\; x{:}\,\Diamond_r y) \longrightarrow u{:}\,\exists x\,\Diamond_r x}\,\forall\text{L}}{\forall x\,\exists y\,(y{:}\,\Diamond_r x \;\&\; x{:}\,\Diamond_r y) \longrightarrow \forall y\, y{:}\,\exists x\,\Diamond_r x}\,\forall\text{R}$$

More generally, we have the following result:

Lemma 7 Any proof in S(M+AΘ+C) can be transformed into a proof in which all cuts have the form

$$\dfrac{\dfrac{}{\Gamma \longrightarrow \Delta, \varphi}\,\Theta \qquad \overset{\pi}{\varphi, \Gamma' \longrightarrow \Delta'}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}$$

where φ is a formula of Θ and is the principal formula of the final rule of π.

Proof This is a standard result for adding axioms to a system of rules from which C is eliminated ([15]). The 'axiom cuts', as they are called, are the only cuts whose rank cannot be reduced by the standard elimination algorithm.

To go further, we need to replace the axioms from Θ with rules that do the same work. In the above example, we need a rule for proving sequents of the form $\Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi$. Working backwards, we can find out what is required:

$$\dfrac{\dfrac{}{\longrightarrow \forall z\,(z{:}\,\exists x\,\varphi \equiv \exists x\, z{:}\,\varphi)}\,\Theta \qquad \dfrac{\dfrac{\dfrac{}{u{:}\,\exists x\,\varphi, \exists x\, u{:}\,\varphi, \Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi}\,\text{I} \qquad \dfrac{\Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi, u{:}\,\exists x\,\varphi, u{:}\,\varphi[{}^{x}_{v}]}{\Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi, u{:}\,\exists x\,\varphi, \exists x\, u{:}\,\varphi}\,\exists\text{R}}{(u{:}\,\exists x\,\varphi \equiv \exists x\, u{:}\,\varphi), \Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi}\,\equiv\text{L}}{\forall z\,(z{:}\,\exists x\,\varphi \equiv \exists x\, z{:}\,\varphi), \Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi}\,\forall\text{L}}{\Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi}\,\text{C}$$

The derivation suggests the rule

$$\Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi, u{:}\,\exists x\,\varphi, u{:}\,\varphi[{}^{x}_{v}] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi$$

But we can do some cleaning up. The rule is derivable from the simpler rule

$$\Gamma \longrightarrow \Delta, u{:}\,\varphi[{}^{x}_{v}] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u{:}\,\exists x\,\varphi$$

which we call $:\exists\text{R}$. Given this rule, our example proof can be rewritten without the cut, as follows:

$$\begin{prooftree}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$v\colon \Diamond_r u, u\colon \Diamond_r v \longrightarrow u\colon \Diamond_r v$}
\RightLabel{$:\exists\text{R}$}
\UnaryInfC{$v\colon \Diamond_r u, u\colon \Diamond_r v \longrightarrow u\colon \exists x\, \Diamond_r x$}
\RightLabel{$\&\text{L}$}
\UnaryInfC{$(v\colon \Diamond_r u \,\&\, u\colon \Diamond_r v) \longrightarrow u\colon \exists x\, \Diamond_r x$}
\RightLabel{$\exists\text{L}$}
\UnaryInfC{$\exists y\, (y\colon \Diamond_r u \,\&\, u\colon \Diamond_r y) \longrightarrow u\colon \exists x\, \Diamond_r x$}
\RightLabel{$\forall\text{L}$}
\UnaryInfC{$\forall x\, \exists y\, (y\colon \Diamond_r x \,\&\, x\colon \Diamond_r y) \longrightarrow u\colon \exists x\, \Diamond_r x$}
\RightLabel{$\forall\text{R}$}
\UnaryInfC{$\forall x\, \exists y\, (y\colon \Diamond_r x \,\&\, x\colon \Diamond_r y) \longrightarrow \forall y\, y\colon \exists x\, \Diamond_r x$}
\end{prooftree}$$

Using the same method, we get a rule for $:\exists$ on the left:

$$:\exists\text{L} \quad u\colon \varphi[^x_v], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \exists x\, \varphi, \Gamma \longrightarrow \Delta$$

with the restriction that v does not occur in φ, Γ, Δ. Together, these two rules will replace the need for the axiom from Θ, with the advantage that they obey a subformula property. (This will be proved later, in Theorem 10.)

We can find similar rules for all the hybrid operators. They are listed in Figure 3. Some of the rules have been split into a Hybrid Rule and an Interface Rule. For example, an analysis of proofs of $\Diamond_r \varphi$ on the left gives us the rule

$$u\colon \Diamond_r \varphi, ruv, v\colon \varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \Diamond_r \varphi, \Gamma \longrightarrow \Delta$$

We split this into the two rules

$:\Diamond_r\text{L}$	$u\colon \Diamond_r v, v\colon \varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \Diamond_r \varphi, \Gamma \longrightarrow \Delta$	Hybrid
$A_2\text{L}$	$ruv, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \Diamond_r v, \Gamma \longrightarrow \Delta$	Interface

from which the general rule can be derived. The advantage of this split is that the Hybrid Rule uses only formulas of the form $u\colon \varphi$. Formulas such as ruv that are not of this form are restricted to applications of the Interface Rules. This gives us a useful tightening of the Subformula Property that we will make good use of shortly.

Let $:H$ be the set of Hybrid Rules, and let $:H/M$ be the set of interface rules. The above discussion motivates the following series of results:

Lemma 8 A sequent of M is a theorem of $\text{S}(M{+}\text{A}\Theta{+}\text{C})$ if and only if it is a theorem of $\text{S}(M{+}{:}H{+}{:}H/M{+}\text{C})$.

Proof We need only show that each of the rules in $:H$ and $:H/M$ is derivable in $\text{S}(M{+}\text{A}\Theta{+}\text{C})$, and that each of the axioms of $\text{A}\Theta$ is derivable in $\text{S}(M{+}{:}H{+}{:}H/M{+}\text{C})$. We have already given an example of a derivation of the first kind; our method of discovering the rules is to construct such a derivation. Here is an example of the derivation of an axiom from a rule. Let π be

$$\begin{prooftree}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$ruv, v\colon \varphi, \Gamma \longrightarrow \Delta, ruv$}
\RightLabel{$A_2\text{L}$}
\UnaryInfC{$u\colon \Diamond_r v, v\colon \varphi, \Gamma \longrightarrow \Delta, ruv$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$u\colon \Diamond_r v, v\colon \varphi, \Gamma \longrightarrow \Delta, v\colon \varphi$}
\RightLabel{$\&\text{R}$}
\BinaryInfC{$u\colon \Diamond_r v, v\colon \varphi, \Gamma \longrightarrow \Delta, (ruv \,\&\, v\colon \varphi)$}
\RightLabel{$\exists\text{R}$}
\UnaryInfC{$u\colon \Diamond_r v, v\colon \varphi, \Gamma \longrightarrow \Delta, \exists x\, (rux \,\&\, x\colon \varphi)$}
\end{prooftree}$$

Hybrid Logical Rules :H	
$:\mathrm{L}_1$	$u\colon v, \Gamma[^w_u] \longrightarrow \Delta[^w_u] \quad \Rightarrow \quad u\colon v, \Gamma[^w_v] \longrightarrow \Delta[^w_v].$
$:\mathrm{L}_2$	$u\colon v, \Gamma[^w_v] \longrightarrow \Delta[^w_v] \quad \Rightarrow \quad u\colon v, \Gamma[^w_u] \longrightarrow \Delta[^w_u].$
$:\mathrm{R}$	$\Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon u$
$:\sim\mathrm{L}$	$\Gamma \longrightarrow \Delta, u\colon \varphi \quad \Rightarrow \quad u\colon {\sim}\varphi, \Gamma \longrightarrow \Delta$
$:\sim\mathrm{R}$	$u\colon \varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon {\sim}\varphi$
$:\vee\mathrm{L}$	$u\colon \varphi, \Gamma \longrightarrow \Delta;\ u\colon \psi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon (\varphi \vee \psi), \Gamma \longrightarrow \Delta$
$:\vee\mathrm{R}$	$\Gamma \longrightarrow \Delta, u\colon \varphi, u\colon \psi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon (\varphi \vee \psi)$
$:\exists\mathrm{L}$	$u\colon \varphi[^x_v], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \exists x\, \varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
$:\exists\mathrm{R}$	$\Gamma \longrightarrow \Delta, u\colon \varphi[^x_v] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon \exists x\, \varphi$
$:\Diamond\mathrm{L}$	$v\colon \varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \Diamond\varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
$:\Diamond\mathrm{R}$	$\Gamma \longrightarrow \Delta, v\colon \varphi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon \Diamond\varphi$
$:\Diamond_r\mathrm{L}$	$u\colon \Diamond_r v, v\colon \varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \Diamond_r\varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
$:\Diamond_r\mathrm{R}$	$\Gamma \longrightarrow \Delta, v\colon \varphi;\ \Gamma \longrightarrow \Delta, u\colon \Diamond_r v \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon \Diamond_r\varphi$
$:@\mathrm{L}$	$v\colon \varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon @_v\varphi, \Gamma \longrightarrow \Delta$
$:@\mathrm{R}$	$\Gamma \longrightarrow \Delta, v\colon \varphi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon @_v\varphi$
$:\downarrow\mathrm{L}$	$u\colon \varphi[^x_u], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \downarrow_x \varphi, \Gamma \longrightarrow \Delta$
$:\downarrow\mathrm{R}$	$\Gamma \longrightarrow \Delta, u\colon \varphi[^x_u] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon \downarrow_x \varphi$

Interface Rules :H/M	
$A_1\mathrm{L}$	$pu, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon p, \Gamma \longrightarrow \Delta$
$A_1\mathrm{R}$	$\Gamma \longrightarrow \Delta, pu \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon p$
$A_2\mathrm{L}$	$ruv, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u\colon \Diamond_r v, \Gamma \longrightarrow \Delta$
$A_2\mathrm{R}$	$\Gamma \longrightarrow \Delta, ruv \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, u\colon \Diamond_r v$

Figure 3: The Hybrid Logical Rules and Interface Rules

Then

$$
\begin{prooftree}
\AxiomC{π}
\RightLabel{$\Diamond_r\mathrm{L}$}
\UnaryInfC{$u\colon \Diamond_r\varphi, \Gamma \longrightarrow \Delta, \exists x\,(rux \,\&\, x\colon \psi)$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$ruv, v\colon \varphi, \Gamma \longrightarrow \Delta, ruv$}
\RightLabel{$A_2\mathrm{R}$}
\UnaryInfC{$ruv, v\colon \varphi, \Gamma \longrightarrow \Delta, u\colon \Diamond_r v$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$ruv, v\colon \varphi, \Gamma \longrightarrow \Delta, v\colon \varphi$}
\RightLabel{$\Diamond_r\mathrm{R}$}
\BinaryInfC{$ruv, v\colon \varphi, \Gamma \longrightarrow \Delta, u\colon \Diamond_r\varphi$}
\RightLabel{$\&\mathrm{L}$}
\UnaryInfC{$(ruv \,\&\, v\colon \varphi), \Gamma \longrightarrow \Delta, u\colon \Diamond_r\varphi$}
\RightLabel{$\exists\mathrm{L}$}
\UnaryInfC{$\exists x\,(rux \,\&\, x\colon \varphi), \Gamma \longrightarrow \Delta, u\colon \Diamond_r\varphi$}
\RightLabel{$\equiv\mathrm{R}$}
\BinaryInfC{$\Gamma \longrightarrow \Delta, (u\colon \Diamond_r\varphi \equiv \exists x\,(rux \,\&\, x\colon \varphi))$}
\RightLabel{$\forall\mathrm{R}$}
\UnaryInfC{$\Gamma \longrightarrow \Delta, \forall z\,(z\colon \Diamond_r\varphi \equiv \exists x\,(rzx \,\&\, x\colon \varphi))$}
\end{prooftree}
$$

The other derivations are all similarly straightforward.

Lemma 9 C is admissible in S(M+:H+:H/M).

Proof To eliminate cuts from proofs in S(M+:H+:H/M+C) we follow the method described in the proof of Theorem 2. We have to show that (1) cuts can be pushed through the new rules when the cut formula is not principal, and (2) that cut rank can be decreased when the cut formula is the principal formula of applications of the new rules. Most of the new rules do not alter the non-principal formulas, and so (1) is straightforward—we omit the details. The only problematic case is that of the Hybrid rules :L$_1$ and :L$_2$, but they are treated in the same way as we treated the =L rules in the proof of Theorem 2.

Again, for most of the rules the transformations required for (2) are exactly analogous to those of their cousins in M. For example, :∨L and :∨R are transformed in the same way as ∨L and ∨R. The transformations required for the Interface Rules are straightforward:

$$\dfrac{\dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, pu}}{\Gamma \longrightarrow \Delta, u{:}\,p}\,A_1\text{R} \qquad \dfrac{\overset{\pi_2}{pu, \Gamma' \longrightarrow \Delta'}}{u{:}\,p, \Gamma' \longrightarrow \Delta'}\,A_1\text{L}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C} \quad \rightsquigarrow \quad \dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, pu} \quad \overset{\pi_2}{pu, \Gamma' \longrightarrow \Delta'}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}$$

The A$_2$ rules are dealt with similarly. The only rules that deserve comment are those for the modal ($\Diamond$, $\Diamond_r$) and hybrid (@, ↓) operators, so we finish the proof with these. First the modals:

$$\dfrac{\dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, v{:}\,\varphi}}{\Gamma \longrightarrow \Delta, u{:}\,\Diamond\varphi}\,\Diamond\text{R} \qquad \dfrac{\overset{\pi_2}{w{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{u{:}\,\Diamond\varphi, \Gamma' \longrightarrow \Delta'}\,\Diamond\text{L}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C} \quad \rightsquigarrow \quad \dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, v{:}\,\varphi} \quad \overset{\pi_2^*[\substack{w\\v}]}{v{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}$$

(As before, π_i^* is the result of renaming the parameters in π_i that are introduced by a restricted rule, such as ∃L. Note that $\Gamma'[\substack{w\\v}] = \Gamma'$ and $\Delta'[\substack{w\\v}] = \Delta'$ because w cannot occur in Γ', Δ'.)

$$\dfrac{\dfrac{\overset{\pi_1'}{\Gamma \longrightarrow \Delta, u{:}\,\Diamond_r v} \quad \overset{\pi_1''}{\Gamma \longrightarrow \Delta, v{:}\,\varphi}}{\Gamma \longrightarrow \Delta, u{:}\,\Diamond_r\varphi}\,\Diamond_r\text{R} \qquad \dfrac{\overset{\pi_2}{u{:}\,\Diamond_r w, w{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{u{:}\,\Diamond_r\varphi, \Gamma' \longrightarrow \Delta'}\,\Diamond_r\text{L}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}$$

$$\rightsquigarrow \quad \dfrac{\overset{\pi_1'}{\Gamma \longrightarrow \Delta, u{:}\,\Diamond_r v} \qquad \dfrac{\overset{\pi_1''}{\Gamma \longrightarrow \Delta, v{:}\,\varphi} \quad \overset{\pi_2^*[\substack{w\\v}]}{u{:}\,\Diamond_r v, v{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{u{:}\,\Diamond_r v, \Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}$$

Now for the hybrid operators:

$$\dfrac{\dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, v{:}\,\varphi}}{\Gamma \longrightarrow \Delta, u{:}\,@_v\varphi}\,@\text{R} \qquad \dfrac{\overset{\pi_2}{v{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{u{:}\,@_v\varphi, \Gamma'\Delta' \longrightarrow}\,@\text{L}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C} \quad \leadsto \quad \dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, v{:}\,\varphi} \quad \overset{\pi_2}{v{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}$$

$$\dfrac{\dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, u{:}\,\varphi}}{\Gamma \longrightarrow \Delta, u{:}\downarrow_x \varphi}\downarrow\text{R} \qquad \dfrac{\overset{\pi_2}{u{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{u{:}\downarrow_x \varphi, \Gamma'\Delta' \longrightarrow}\downarrow\text{L}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C} \quad \leadsto \quad \dfrac{\overset{\pi_1}{\Gamma \longrightarrow \Delta, u{:}\,\varphi} \quad \overset{\pi_2}{u{:}\,\varphi, \Gamma' \longrightarrow \Delta'}}{\Gamma, \Gamma' \longrightarrow \Delta, \Delta'}\,\text{C}$$

Theorem 10 A sequent of M is Θ-valid if and only if it is a theorem of S(M+:H + :H/M).

Proof Soundness (the 'if' direction) follows from the fact that each of the rules in :H and :H/M is derivable in S(M+AΘ+C), which is sound by Theorem 6. For completeness, suppose that $\Gamma \longrightarrow \Delta$ is Θ-valid. By Theorem 6, there is a proof of this sequent in S(M+AΘ+C), which can be converted into a proof in S(M+:H+:H/M+C) by Lemma 8, and then into a proof without C, by Lemma 9.

The calculus S(M+:H+:H/M) gives a complete characterization of the valid sequents of M and comes very close to having the Subformula Property. The rules of SM have the Subformula Property, as do those of :H, with a suitable definition of the subformulas of $u{:}\,\varphi$. We say that $v{:}\,\psi$ is a *subformula* of $u{:}\,\varphi$ if ψ is the result of replacing zero or more occurrences of the parameters of a (genuine) subformula of φ. The Interface Rules lack the Subformula Property, but this is not too serious; we can replace them with axioms that have it (see Figure 4).

Interface Axioms A:H/M		
A_1^*L	$\Rightarrow$	$u{:}\,p, \Gamma \longrightarrow \Delta, pu$
A_1^*R	$\Rightarrow$	$pu, \Gamma \longrightarrow \Delta, u{:}\,p$
A_2^*L	$\Rightarrow$	$u{:}\,\Diamond_r v, \Gamma \longrightarrow \Delta, ruv$
A_2^*R	$\Rightarrow$	$ruv, \Gamma \longrightarrow \Delta, u{:}\,\Diamond_r v$

Figure 4: Interface Axioms A:H/M

Corollary 11 A sequent of M is Θ-valid if and only if it is a theorem of S(M+:H + A:H/M).

Proof From Theorem 10. We have to show that applications of the Interface Rules can be replaced by applications of the Interface Axioms. This is simply a matter of showing that the Interface Rules commute with all the other rules until they get to the leaves of the proof tree. All leaves end in axioms: I,=R,: R, and the Interface

Axioms. In each case, the Interface Rule can be eliminated using another axiom. For example,

$$\dfrac{\dfrac{}{pu,\Gamma \longrightarrow \Delta, pu}\,\mathrm{I}}{u\colon p,\Gamma \longrightarrow \Delta, pu}\,A_1\mathrm{L} \quad \leadsto \quad \dfrac{}{u\colon p,\Gamma \longrightarrow \Delta, pu}\,A_1^*\mathrm{L}$$

$$\dfrac{\dfrac{}{u\colon \Diamond_r v,\Gamma \longrightarrow \Delta, ruv}\,\mathrm{A}_2^*\mathrm{L}}{u\colon \Diamond_r v,\Gamma \longrightarrow \Delta, u\colon \Diamond_r v}\,A_2\mathrm{R} \quad \leadsto \quad \dfrac{}{u\colon \Diamond_r v,\Gamma \longrightarrow \Delta, u\colon \Diamond_r v}\,\mathrm{I}$$

The system S(M+:H+A:H/M) has the Subformula Property and so we have reached the goal of this section.

5 Internalization and Rules For All

Now that Θ-validity in M has been captured with a respectable sequent calculus, we can use Lemma 1 to provide a calculus for validity in H. Let S:H be the set consisting of the Structural Rules, I and S, together with the Hybrid Rules :H.

Theorem 12 Suppose that u does not occur in Γ, Δ. The sequent $\Gamma \longrightarrow \Delta$ of H is valid if and only if $u\colon \Gamma \longrightarrow u\colon \Delta$ is a theorem of S:H. Moreover, the proof uses only the rules for the operators occurring in Γ, Δ.

Proof By Corollary 11, the calculus S(M+S:H+A:H/M) is sufficient, but this calculus enjoys the Subformula Property, and so any proof of a sequent of the form $u\colon \Gamma \longrightarrow u\colon \Delta$ uses only the rules of S:H.

This is a great improvement on S(M+AΘ+C), but still slightly unsatisfactory because the proofs of S:H are not fully internalized. The formulas occurring in the proof are not formulas of H; they are all of the form $u\colon \varphi$, which belongs to the formal metalanguage M.

To push internalization as far as possible, we will reformulate the rules of S:H using the hybrid operator @. This gives us the system S@H, shown in Figure 5. Without @ we would have to look much more closely at the structure of proofs to make any further progress toward internalization.
S@H is fully internalized: the only formulas occurring in proofs are formulas of H. What's more, it has the Subformula Property, if we reinterpret the definition of subformula appropriately. The only remaining drawback is that the calculus only applies to a fragment of the language: the formulas of the form $@_u\varphi$. This is sufficient for the purpose of characterizing validity, because every sequent $\Gamma \longrightarrow \Delta$ is equivalent to a sequent in this fragment, namely, $@_u\Gamma \longrightarrow @_u\Delta$. Yet the absence of rules for dealing with sequents without @ is regrettable, and unnecessary. In the final tuning of our proof-theoretic apparatus, we aim for a more egalitarian logic in which there are 'Rules for All'.

Nominals—individual variables occurring as formulas—play an essential part in liberating the calculus from @-prefixed formulas. A single nominal parameter u on the left side of a sequent is enough to anchor all non-@-prefixed formulas to the same element and so removes"' the need for them to share a prefix. To shift between @-prefixed formulas and free nominals we need the Nominal Rules N, shown in Figure 6.

Internalized Hybrid Logical Rules $@H$	
@L$_1$	$@_u v, \Gamma[{}^{w}_{u}] \longrightarrow \Delta[{}^{w}_{u}] \quad \Rightarrow \quad @_u v, \Gamma[{}^{w}_{v}] \longrightarrow \Delta[{}^{w}_{v}].$
@L$_2$	$@_u v, \Gamma[{}^{w}_{v}] \longrightarrow \Delta[{}^{w}_{v}] \quad \Rightarrow \quad @_u v, \Gamma[{}^{w}_{u}] \longrightarrow \Delta[{}^{w}_{u}].$
@R	$\Rightarrow \quad \Gamma \longrightarrow \Delta, @_u u$
@∼L	$\Gamma \longrightarrow \Delta, @_u\varphi \quad \Rightarrow \quad @_u{\sim}\varphi, \Gamma \longrightarrow \Delta$
@∼R	$@_u\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, @_u{\sim}\varphi$
@∨L	$@_u\varphi, \Gamma \longrightarrow \Delta;\ @_u\psi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad @_u(\varphi \vee \psi), \Gamma \longrightarrow \Delta$
@∨R	$\Gamma \longrightarrow \Delta, @_u\varphi, @_u\psi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, @_u(\varphi \vee \psi)$
@∃L	$@_u\varphi[{}^{x}_{v}], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad @_u\exists x\,\varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
@∃R	$\Gamma \longrightarrow \Delta, @_u\varphi[{}^{x}_{v}] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, @_u\exists x\,\varphi$
@◇L	$@_v\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad @_u\Diamond\varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
@◇R	$\Gamma \longrightarrow \Delta, @_v\varphi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, @_u\Diamond\varphi$
@◇$_r$L	$@_u\Diamond_r v, @_v\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad @_u\Diamond_r\varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
@◇$_r$R	$\Gamma \longrightarrow \Delta, @_v\varphi;\ \Gamma \longrightarrow \Delta, @_u\Diamond_r v \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, @_u\Diamond_r\varphi$
@@L	$@_v\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad @_u@_v\varphi, \Gamma \longrightarrow \Delta$
@@R	$\Gamma \longrightarrow \Delta, @_v\varphi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, @_u@_v\varphi$
@ ↓L	$@_u\varphi[{}^{x}_{u}], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad @_u \downarrow_x \varphi, \Gamma \longrightarrow \Delta$
@ ↓R	$\Gamma \longrightarrow \Delta, @_u\varphi[{}^{x}_{u}] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, @_u \downarrow_x \varphi$

Figure 5: Internalized Hybrid Logical Rules $@H$

Lemma 13 A sequent of H is valid if and only if it is a theorem of S($@H$+N).

Proof First note that a sequent $\Gamma \longrightarrow \Delta$ of H is valid if and only if $@_u\Gamma \longrightarrow @_u\Delta$ is a theorem of S$@H$ for u not in Γ, Δ. This follows from Theorem 12 together with the observation that the rules, and hence the proofs, in these two systems are isomorphic. We can convert one to the other simply by replacing u: by $@_u$ and vice versa. But now suppose that $\Gamma \longrightarrow \Delta$ is valid and so $@_u\Gamma \longrightarrow @_u\Delta$ is a theorem of S$@H$ (for u not in Γ, Δ). Weakening, we get $u, @_u\Gamma \longrightarrow @_u\Delta$, from which we can derive $u, \Gamma \longrightarrow \Delta$ using multiple applications of ^@L and ^@R. Finally, we get $\Gamma \longrightarrow \Delta$ using name. For the converse, we need only check that each rule of N is sound.

Now, with the Nominal Rules in place, the rules of $@H$ can be converted to use the nominal-based method of context sharing. We convert each rule of $@H$ of the form

$$@_u\Gamma_1, \Gamma \longrightarrow \Delta, @_u\Delta_1 \quad \Rightarrow \quad @_u\Gamma_2, \Gamma' \longrightarrow \Delta', @_u\Delta_2$$

to the rule

$$u, \Gamma_1, \Gamma \longrightarrow \Delta, \Delta_1 \quad \Rightarrow \quad u, \Gamma_2, \Gamma' \longrightarrow \Delta', \Delta_2$$

If u does not occur in (the general statement of) this rule except in the place we have shown, we can go one step further and convert the rule to

$$\Gamma_1, \Gamma \longrightarrow \Delta, \Delta_1 \quad \Rightarrow \quad \Gamma_2, \Gamma' \longrightarrow \Delta', \Delta_2$$

Nominal Rules N	
$^{\vee}$@L	$u, \varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u, @_u\varphi, \Gamma \longrightarrow \Delta$
$^{\vee}$@R	$u, \Gamma \longrightarrow \Delta, \varphi \quad \Rightarrow \quad u, \Gamma \longrightarrow \Delta, @_u\varphi$
$^{\wedge}$@L	$u, @_u\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u, \varphi, \Gamma \longrightarrow \Delta$
$^{\wedge}$@R	$u, \Gamma \longrightarrow \Delta, @_u\varphi \quad \Rightarrow \quad u, \Gamma \longrightarrow \Delta, \varphi$
name	$u, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma \longrightarrow \Delta$ if u does not occur in Γ, Δ.
term	$u, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma \longrightarrow \Delta$ if all formulas in Γ, Δ are @-prefixed.

Figure 6: Nominal Rules N

(Here the shared element of the non-@-prefixed formulas is given implicitly as the point of evaluation.) This gives us the Nominal-based Internalized Hybrid Logic Rules NH, shown in Figure 7.

Nominal-based Internalized Hybrid Logic Rules NH	
NL	$u, v, \Gamma[^{w}_{u}] \longrightarrow \Delta[^{w}_{u}] \quad \Rightarrow \quad u, v, \Gamma[^{w}_{v}] \longrightarrow \Delta[^{w}_{v}]$.
$\sim$L	$\Gamma \longrightarrow \Delta, \varphi \quad \Rightarrow \quad {\sim}\varphi, \Gamma \longrightarrow \Delta$
$\sim$R	$\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, {\sim}\varphi$
$\vee$L	$\varphi, \Gamma \longrightarrow \Delta;\ \psi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad (\varphi \vee \psi), \Gamma \longrightarrow \Delta$
$\vee$R	$\Gamma \longrightarrow \Delta, \varphi, \psi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, (\varphi \vee \psi)$
$\exists$L	$\varphi[^{x}_{v}], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \exists x\, \varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
$\exists$R	$\Gamma \longrightarrow \Delta, \varphi[^{x}_{v}] \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, \exists x\, \varphi$
$\Diamond$L	$@_v\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Diamond\varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
$\Diamond$R	$\Gamma \longrightarrow \Delta, @_v\varphi \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, \Diamond\varphi$
$\Diamond_r$L	$\Diamond_r v, @_v\varphi, \Gamma \longrightarrow \Delta \quad \Rightarrow \quad \Diamond_r\varphi, \Gamma \longrightarrow \Delta$ if v does not occur in φ, Γ, Δ.
$\Diamond_r$R	$\Gamma \longrightarrow \Delta, @_v\varphi;\ \Gamma \longrightarrow \Delta, \Diamond_r v \quad \Rightarrow \quad \Gamma \longrightarrow \Delta, \Diamond_r\varphi$
$N{\downarrow}$L	$u, \varphi[^{x}_{u}], \Gamma \longrightarrow \Delta \quad \Rightarrow \quad u, \downarrow_x \varphi, \Gamma \longrightarrow \Delta$
$N{\downarrow}$R	$u, \Gamma \longrightarrow \Delta, \varphi[^{x}_{u}] \quad \Rightarrow \quad u, \Gamma \longrightarrow \Delta, \downarrow_x \varphi$

Figure 7: Nominal-based Internalized Hybrid Logic Rules NH

After conversion several of the rules become redundant. The result of converting @R is just a special case of I, and the @@ rules become entirely trivial. These have been omitted from the above table.

Many of the rules in NH are very familiar. The operators of classical logic involve no hybrid interaction and so have returned to a familiar form. Interestingly, the rules for the modal operators use both nominals and the @ operator.

Theorem 14 A sequent of H is valid if and only if it is a theorem of S($N + NH$).

Proof We show that any proof in S($N + NH$) can be converted into a proof in

S(@$H + N$) and vice versa. The theorem follows from Lemma 13. Each rule

$$R \quad @_u\Gamma_1, \Gamma \longrightarrow \Delta, @_u\Delta_1 \quad \Rightarrow \quad @_u\Gamma_2, \Gamma' \longrightarrow \Delta', @_u\Delta_2$$

of @H is first converted into the rule

$$NR \quad u, \Gamma_1, \Gamma \longrightarrow \Delta, \Delta_1 \quad \Rightarrow \quad u, \Gamma_2, \Gamma' \longrightarrow \Delta', \Delta_2$$

Wherever NR occurs in a proof, it can be replaced by the following derivation from S(@$H + N$):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{u, \Gamma_1, \Gamma \longrightarrow \Delta, \Delta_1}{u, @_u\Gamma_1, @_u\Gamma \longrightarrow @_u\Delta, @_u\Delta_1}\ {}^{\vee}@\mathrm{L}{}^{\vee}@\mathrm{R}}{@_u\Gamma_1, @_u\Gamma \longrightarrow @_u\Delta, @_u\Delta_1}\ \text{term}}{@_u\Gamma_2, @_u\Gamma' \longrightarrow @_u\Delta', @_u\Delta_2}\ R}{u, @_u\Gamma_2, @_u\Gamma' \longrightarrow @_u\Delta', @_u\Delta_2}\ \mathrm{W}}{u, \Gamma_2, \Gamma' \longrightarrow \Delta', \Delta_2}\ {}^{\wedge}@\mathrm{L}{}^{\wedge}@\mathrm{R}$$

(The step using R is okay because for all the rules in S@H, the prefixing of $@_u$ to the Γ, Δ and Γ', Δ' does not alter the applicability of the rule.) Similarly, wherever R occurs in a proof, it can be replaced by the following derivation from S($NH + N$), with v new to the proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{@_u\Gamma_1, \Gamma \longrightarrow \Delta, @_u\Delta_1}{v, @_u\Gamma_1, @_v\Gamma \longrightarrow @_v\Delta, @_u\Delta_1}\ \mathrm{W}, {}^{\vee}@\mathrm{L}, {}^{\vee}@\mathrm{R}}{@_u\Gamma_1, @_v\Gamma \longrightarrow @_v\Delta, @_u\Delta_1}\ \text{term}}{u, \Gamma_1, @_v\Gamma \longrightarrow @_v\Delta, \Delta_1}\ \mathrm{W}, {}^{\wedge}@\mathrm{L}, {}^{\wedge}@\mathrm{R}}{u, \Gamma_2, @_v\Gamma \longrightarrow @_v\Delta, \Delta_2}\ NR}{u, @_u\Gamma_2, @_v\Gamma \longrightarrow @_v\Delta, @_u\Delta_2}\ {}^{\vee}@\mathrm{L}{}^{\vee}@\mathrm{R}}{@_u\Gamma_2, @_v\Gamma \longrightarrow @_v\Delta, @_u\Delta_2}\ \text{term}}{v, @_u\Gamma_2, \Gamma \longrightarrow \Delta, @_u\Delta_2}\ \mathrm{W}, {}^{\wedge}@\mathrm{L}, {}^{\wedge}@\mathrm{R}}{@_u\Gamma_2, \Gamma \longrightarrow \Delta, @_u\Delta_2}\ \text{name}$$

(Again, the application of NR is okay because these rules are similarly immune to prefixing the Γ, Δ part.) Finally, in the case that u does not occur in Γ, Δ we need to show that $\Gamma \longrightarrow \Delta$ can be derived from $u, \Gamma \longrightarrow \Delta$ and vice versa. But this is just W in one direction and name in the other.

The price of implementing our policy of Rules for All is that the calculus no longer enjoys the Subformula Property. A proof may contain any number of @-prefixes not in the end sequent, introduced using the ${}^{\wedge}@$ rules. Of course, we know that excessive introduction of prefixes is unnecessary, and from the proofs of Lemma 13 and Theorem 14 we see that only one layer of prefixes is ever needed.

The internalization strategy, illustrated here for hybrid logics, can be applied to a wide range of logical operators. Any first-order definable operator can be tackled in this way, although it is presently unclear to me how far one can go in any given case. For a candidate operator with a first-order definition, it is easy to find rules using the technique shown on page 10. Yet this process is not entirely automated. There is still a little 'tidying up' to be done, and further work showing that Cut can be eliminated. Some problems arise with operators having nested quantifiers because of the need to keep track of dependant variables, but the limits of the method have not yet been established.

For hybrid logics, full internalization is possible only because of the expressive power of the language, specifically the fine control exercised by @ and the nominals. For @-less fragments and other modal logics, there may be no way of duplicating the function of the metalogical : within the object language. In that case, the internalization strategy will halt, awaiting a more specific analysis of the structure of proofs, from which it may be possible to guess restrictions of the kind used in the sequent calculi for S4 and intuitionistic logic. Outside the realm of normal model logics and their hybrid extensions, the boundaries are even less clear. Some preliminary success has been achieved using internalization to construct a Tableaux system for S1 based on its neighbourhood semantics (see[9]).

Acknowledgments

The present paper grew out of an earlier manuscript [14] in which the calculus presented here was shown to have a number of nice properties, and which was subsequently discussed extensively in [3]. The manuscript was hand-written during a short stay at Patrick Blackburn's house in the Lorraine, and then photocopied and distributed to a number of interested people. It has not been published or even converted to electronic form, although it is mentioned in a number of places, including [10], the Hybrid Logics' Home Page. A longer history traces the work back to [11], [12], and [13]. My thanks to Patrick for his hospitality and enthusiasm and to the referees of the paper for useful comments and corrections.

References

[1] C. Areces, P. Blackburn, and M. Marx. The computational complexity of hybrid temporal logics. *Logic Journal of the IGPL*, 8(5):653–679, 2000.

[2] C. Areces, P. Blackburn, and M. Marx. Hybrid logics: Characterization, interpolation and complexity. *Journal of Symbolic Logic*, 2000. To appear.

[3] P. Blackburn. Internalizing labelled deduction. *Journal of Logic and Computation*, 10:136–168, 2000.

[4] P. Blackburn and J. Seligman. Hybrid languages. *Journal of Logic, Language and Information*, 4(3):251–272, 1995. Special issue on decompositions of first-order logic.

[5] P. Blackburn and J. Seligman. What are hybrid languages? In M. Kracht, M. de Rijke, H. Wansing, and M. Zakharyaschev, editors, *Advances in Modal Logic*, volume 1, pages 41–62. CSLI Publications, Stanford University, 1998.

[6] P. Blackburn and M. Tzakova. Hybrid languages and temporal logic. *Logic Journal of the IGPL*, 7(1):27–54, 1999.

[7] S. Demri. Sequent calculi for nominal tense logics: a step towards mechanization? In N. Murray, editor, *Conference on Tableaux Calculi and Related Methods (TABLEAUX), Saratoga Springs, USA*, volume 1617 of *LNAI*, pages 140–154. Springer Verlag, 1999.

[8] S. Demri and R. Goré. Cut-free display calculi for nominal tense logics. In N. Murray, editor, *Conference on Tableaux Calculi and Related Methods (TABLEAUX), Saratoga Springs, USA*, volume 1617 of *LNAI*, pages 155–170. Springer Verlag, 1999.

[9] R. Girle and J. Seligman. In the neighbourhood. Manuscript, 2000.

[10] Hybrid logics' home page. `http://www.hylo.net`.

[11] J. Seligman. A cut-free sequent calculus for elementary situated reasoning. Technical Report HCRC-RP 22, HCRC, Edinburgh, 1991.

[12] J. Seligman. Situated consequence in elementary situation theory. Technical Report IULG-92-16, IULG, Indiana University, 1992.

[13] J. Seligman. The logic of correct description. In M. de Rijke, editor, *Advances in Intensional Logic*, pages 107–135. Kluwer, 1997.

[14] J. Seligman. Proof theory for contextual reasoning. Manuscript, 1998.

[15] A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*, volume 43 of *Tracts in Theoretical Computer Science*. Cambridge, 1997.

[16] M. Tzakova. Tableaux calculi for hybrid logics. In N. Murray, editor, *Proceedings of the Conference on Tableaux Calculi and Related Methods (TABLEAUX), Saratoga Springs, USA*, volume 1617 of *LNAI*, pages 278–292. Springer Verlag, 1999.

Relation Algebra with Binders

Maarten Marx

Applied Logic Lab, ILLC, Universiteit van Amsterdam,
Plantage Muidergracht 24, 1018 TV, Amsterdam, The Netherlands
Email: marx@science.uva.nl

Abstract

The language of relation algebras is expanded with variables denoting individual elements in the domain and with the $\downarrow$ binder from hybrid logic. Every elementary property of binary relations is expressible in the resulting language, something which fails for the relation algebraic language. That the new language is natural for speaking about binary relations is indicated by the fact that both Craig's Interpolation, and Beth's Definability theorems hold for its set of validities. The paper contains a number of worked out examples.

Keywords: binary relations, relation algebras, modal logic, hybrid logic, fork algebras, interpolation, Beth definability.

1 Introduction and Motivation

Tarski wrote

> [...] the calculus of relations has an intrinsic charm and beauty which makes it a source of intellectual delight to all who become acquainted with it. [10: p89]

Still, first-order logic is the language universally understood in our field. We write $\forall x \forall y (R(x,y) \to R(y,x))$ instead of $R \subseteq R^{-1}$, we write $\forall x \forall y \forall z ((R(x,y) \wedge R(y,z)) \to R(x,z))$ instead of $R \circ R \subseteq R$, and $\forall x \forall y \forall z ((R(x,y) \wedge R(x,z)) \to \exists w (R(y,w) \wedge R(z,w)))$ when $R^{-1} \circ R \subseteq R \circ R^{-1}$ expresses the same fact.

I do not want to go into the reasons for the supremacy of first-order logic over relation algebras. I conjecture such research would be both speculative and necessarily interdisciplinary. It might even be that matters boil down to the same reasons as why we are still typing on QWERTY keyboards. Whatever the reasons, I think Tarski himself was not a great advocate of his beloved system.

There is though an obvious reason for preferring first-order logic: not every elementary statement about individuals and binary relations can be expressed in relation algebraic terms. The first counterexample —"the domain contains at least four elements"— was already discovered in 1915 (see Example 3 below). The question whether a given elementary statement can be equivalently expressed in relation algebraic terms is in general undecidable [7, 11]. On the other hand, the set of relation algebraic terms can be characterized as a fragment of first-order logic: every relation algebraic term is equivalent to a first-order formula with at most two free variables and in which at most 3 variables may occur and vice versa [11].

Tarski knew when he wrote his delightful sentence quoted above that under some mild conditions (the most important is logicality of the operators) no finite expansion of the relation algebraic language could provide full first-order expressivity. A finite expansion with a non-logical operator (called fork) was made in fork algebras (cf. the survey containing an extensive bibliography [5]). The fork operation takes two binary relations and produces a new binary relation. The fork operation ∇ is defined assuming that the universe is closed under taking pairs of elements: for R, S binary relations and x, y elements of the domain

$$xR\nabla Sy \iff \exists u \exists v(y = \langle u, v\rangle \wedge xRu \wedge xSv).$$

In this paper, I will define a much simpler but infinite expansion instead. A number of examples show that the language is natural and easy to use. The expansion has full first-order expressivity. As an immediate result of this the language has interpolation and definability theorems. Both features are lacked by the relation algebraic language. These two properties are powerful indicators of the naturalness of a language, in particular of a good fit between syntax and semantics.

The expansion proposed here is not new. The same expansion has been applied in modal logic under the name of hybrid logic. As hybrid logic is well motivated in this issue and elsewhere [3], a brief explanation suffices. Hybrid logic $\mathcal{H}(@, \downarrow)$ adds three features to the standard uni-modal propositional language:

1. A set VAR of new atomic formulas. Elements of this set are variables ranging over the set of states of a Kripke model. The variables obtain a value by means of an assignment function. A variable x holds at a state s in Kripke model $\mathfrak{M}$ under an assignment g (notation: $\mathfrak{M}, s, g \Vdash x$) iff g assigns the state s to x.

2. For every $x \in VAR$ and for every formula φ, $@_x\varphi$ is a formula too. The formula $@_x\varphi$ is a "holds" predicate: it states that at the state to which x is bound, φ holds. Formally:
$$\mathfrak{M}, s, g \Vdash @_x\varphi \iff \mathfrak{M}, g(x), g \Vdash \varphi.$$

3. For every $x \in VAR$ and formula φ, $\downarrow x.\varphi$ is a formula too. The formula $\downarrow x.\varphi$ binds the variable x to the current state of evaluation:
$$\mathfrak{M}, s, g \Vdash \downarrow x.\varphi \iff \mathfrak{M}, s, g^s_x \Vdash \varphi,$$
in which g^s_x is the assignment which is just like g except that x is sent to s.

   The downarrow binder provides the means to temporarily store the state of evaluation in the value of the variable x. The operator $\downarrow x$ obeys the same binding rules as the quantifiers in first-order logic. The binder $\downarrow x$ is best seen as an explicit substitution operator. This can readily be seen from its standard translation [1] into first-order logic: $ST_x(\downarrow x_i.\varphi) = (ST_x(\varphi))[x_i/x]$. Note that $(ST_x(\varphi))[x_i/x] \equiv \exists x_i(x_i = x \wedge ST_x(\varphi)) \equiv \forall x_i(x_i = x \rightarrow ST_x(\varphi))$.

A nice example of the strength of the formalism is given by the following definition of the until operator. (a) gives the intended meaning of Until and (b) its definition in hybrid logic.

$$\begin{array}{llcll}
(a) & \mathfrak{M}, x \Vdash \mathsf{Until}(\varphi,\psi) & \iff & \exists y(xRy \wedge \mathfrak{M}, y \Vdash \varphi & \wedge \quad \forall z(xRzRy \rightarrow \mathfrak{M}, z \Vdash \psi)) \\
(b) & \mathsf{Until}(\varphi,\psi) & := & \downarrow x.(\Diamond \downarrow y.(\varphi & \wedge \quad @_x\Box(\Diamond y \rightarrow \psi))).
\end{array}$$

That is, we name the current state x, use $\Diamond$ to move to an accessible φ-state which we name y, and then use $@_x$ to jump back to the state named x. We then insist that ψ holds in all successors of the x named state which precede the state named y.

A second example is the spy-point formula from [1]:

$$\downarrow z.\Box\Box \downarrow x.@_z\Diamond x. \qquad (1)$$

If it is true in a model at a state s, then every state in the submodel generated from s is reachable in one R step from s. The state s is "spying" on the model. (1) is an example of a *pure sentence*, a formula without propositional variables and free occurrences of variables. Pure sentences can be used to force properties of the accessibility relation. For instance, the following two statements are equivalent on every Kripke model $\mathfrak{M}$:

1. $\downarrow z.\Box\Box \downarrow x.@_z\Diamond x$ holds at every state in $\mathfrak{M}$,

2. the accessibility relation R in $\mathfrak{M}$ is transitive.

The hybrid logic has a nice characterization as the bounded fragment of first-order logic. Like first-order logic it is undecidable and has interpolation and definability properties [1].

I now turn to the hybridization of relation algebras.

2 The Expansion

Relation algebraic terms are defined from atomic relation symbols and the special relation symbols $\emptyset, \top$ and $1'$ by the following grammar:

$$R \cap S \mid R \cup S \mid -R \mid R^{-1} \mid R \circ S.$$

The inequality relation $-1'$ is abbreviated as $0'$. Models are just first-order models (D, I) consisting of a domain of individuals D and an interpretation I of the atomic relation symbols. As the language contains only binary relations it holds that $I(R) \subseteq D \times D$ for every atomic symbol R. In a model $\mathfrak{M} = (D, I)$, every relation algebraic term denotes a binary relation. $\mathfrak{M} \models a[R]b$ denotes that the pair (a, b) stands in the relation R in model $\mathfrak{M}$. If the model is clear from the context just $a[R]b$ is written. The statement $\mathfrak{M} \models R = S$ expresses that in $\mathfrak{M}$, R and S denote the same relation.

The meaning of the complex terms is defined inductively as follows. For atomic R, $\mathfrak{M} \models a[R]b$ iff $(a, b) \in I(R)$. Relations constructed with the Boolean connectives get the standard meaning with $\top$ denoting the universal relation $D \times D$. Then $\mathfrak{M} \models a[1']b \iff a = b$, $\mathfrak{M} \models a[R^{-1}]b \iff \mathfrak{M} \models b[R]a$ and $\mathfrak{M} \models a[R \circ S]b$ iff there exists a $c \in D$ such that $\mathfrak{M} \models a[R]c$ and $\mathfrak{M} \models c[S]b$.

An important defined operator is the residual $\backslash$. It can be viewed as a dynamic implication, witness the validity of the inequality $R{\circ}(R\backslash S) \subseteq S$. The $\backslash$ is term-definable using negation, converse and composition:

$$R\backslash S \equiv -(R^{-1}{\circ}{-S}).$$

With this definition it holds that

$$a[R\backslash S]b \iff \text{for all } c \in D, \text{ if } c[R]a, \text{ then } c[S]b.$$

The binding machinery from the hybrid logic $H(\downarrow, @)$ is now added to the relation algebraic language. Let VAR be a countably infinite set of variables. Add to the construction rules of the relation algebraic terms the following two:

- all elements in VAR are terms, and
- if $x \in VAR$ and R is a term, then $\downarrow^0_x.R$ is a term as well.

The newly obtained language is denoted by $RL\!\downarrow$, pronounced as "RL downarrow". Assignment functions are needed to give meaning to the new terms. For $\mathfrak{M} = (D, I)$ a model, let $g : VAR \longrightarrow D$ be an assignment. In the definition of the denotation of a term R in $\mathfrak{M}$ relative to g (in symbols: $\mathfrak{M} \models a[R]_g b$), the old clauses do not change apart from adding the assignment everywhere. The new terms are interpreted as follows:

$$\begin{array}{lcl} \mathfrak{M} \models a[x]_g b & \iff & g(x) = a = b, \text{ for } x \in VAR \\ \mathfrak{M} \models a[\downarrow^0_x.R]_g b & \iff & \mathfrak{M} \models a[R]_{g^a_x} b, \end{array}$$

in which g^a_x is the assignment which is just like g except that x is sent to a.

Note that the term x, when interpreted on a model with assignment g, denotes the binary relation $\{(g(x), g(x))\}$. These variables work just as pronouns in natural language and can be freely combined with other relation symbols. A natural language sentence like *"He loves her"* can be translated as $x \circ \mathsf{love} \circ y$. Now $x \circ \mathsf{love} \circ y$ is true on a model under an assignment g precisely when $(g(x), g(y))$ are in the interpretation of the relation love. *He loves someone who loves him* is naturally translated as $x \circ \mathsf{love} \circ \mathsf{love} \circ x$.

An $RL\!\downarrow$ term is called closed if all variables occurring in it are bound by a downarrow.

In writing formulas, the following abbreviations are sometimes useful:

$$\begin{array}{lcl} \downarrow^1_x.R & \equiv & (\downarrow^0_x.R^{-1})^{-1} \\ @^0_x R & \equiv & \top{\circ}x{\circ}R \\ @^1_x R & \equiv & R{\circ}x{\circ}\top. \end{array}$$

Writing out the definitions, we obtain that

$$\begin{array}{lcl} \mathfrak{M} \models a[\downarrow^1_x.R]_g b & \iff & \mathfrak{M} \models a[R]_{g^b_x} b \\ \mathfrak{M} \models a[@^0_x.R]_g b & \iff & \mathfrak{M} \models g(x)[R]_g b \\ \mathfrak{M} \models a[@^1_x.R]_g b & \iff & \mathfrak{M} \models a[R]_g g(x). \end{array}$$

The new language looks very much like $\mathcal{H}(@, \downarrow)$. Because terms denote binary relations, there are two different versions of both $\downarrow$ and @, one for each coordinate.

Remark 1 A different and maybe more natural way of adding the possibility of naming elements of the domain is to add instead of the $\downarrow^0_x$ operator, the following form of composition:

- if $x \in VAR$ and R and S are terms, then also $R{\circ_x}S$ is a term.

$R{\circ_x}S$ gets the following meaning

$$\mathfrak{M} \models a[R{\circ_x}S]_g b \quad\iff\quad (\exists c \in D) : \mathfrak{M} \models a[R]_{g^c_x} c \text{ and } \mathfrak{M} \models c[S]_{g^c_x} b.$$

Note that $\circ_x$ binds free occurrences of x in $R \circ_x S$ *both* in R and in S. This backwards binding power is solely due to the fact that we write $\circ_x$ as an infix operator. Thus $R{\circ_x}S$ stores the value of the intermediate point of the composition in the value of x. It does not matter which of $\circ_x$ and $\downarrow^0_x$ is taken as primitive because these two connectives are interdefinable:

$$R{\circ_x}S = \downarrow^1_x.R\circ \downarrow^0_x.S \text{ and } \downarrow^0_x.R = 1'{\circ_x}R.$$

I have chosen for the downarrow binders because I found them easier to use.[1]

3 Examples

A number of examples are presented. All the properties expressed in $RL\!\downarrow$ are first-order properties not expressible in relation algebraic terms. All examples except the one about siblings can be found in [11], section 3.6.

Example 1 Express that any two elements have a greatest lower bound (in the order established by E):

$$\forall x \forall y \exists z (zEx \wedge zEy \wedge \forall u((uEx \wedge uEy) \rightarrow uEz)). \tag{2}$$

(2) naturally breaks down into two parts:

1. the existential part providing a lower bound. This is easily expressed by $E^{-1}{\circ}E$. In a picture:

$$\begin{tikzcd} & \cdot \arrow[dl, "E"'] \arrow[dr, "E"] & \\ x \cdot \arrow[rr] & & \cdot\, y \end{tikzcd}$$

2. and the universal part making it the greatest lower bound. This is more difficult to express. In a picture it says that the following constellation is forbidden if z is the intended greatest lower bound.

[1]A more radical proposal (also suggested by two of the referees) is to have variables referring to *pairs* of elements in the domain, and a corresponding binder. Obviously such an expansion is also term-definably equivalent to the more conservative one developed here. See also Example 4 below in which such a pair-variable can be used.

$$\begin{tikzcd}
 & u \arrow[dl, "E"'] \arrow[dr, "E"] \arrow[d, "-E"] & \\
 & z \arrow[dl] \arrow[dr] & \\
x \arrow[rr] & & y
\end{tikzcd}$$

If we could use the term x as a relation which holds only for the pair (x, x) we can express this as follows: the pair (z, y) should not be in the relation

$$(-E \cap E{\circ}\mathsf{x}{\circ}\top)^{-1}{\circ}E \tag{3}$$

By definition of the residual, (z, y) is not in the relation expressed by (3) iff (z, y) stands in the relation

$$(-E \cap E{\circ}\mathsf{x}{\circ}\top)\backslash{-E}. \tag{4}$$

It now follows that (5) expresses that any two elements have a greatest lower bound

$$\downarrow^0_{\mathsf{x}}.(E^{-1}{\circ}(E \cap (4))) = \top. \tag{5}$$

(Note: in order to distinguish the x used as a name for a point in the drawings and the variable x in the formulas, the variables are written in sans serif script in these examples.)

Example 2 The next example from [11] which is not expressible in ordinary relation algebra is the union axiom:

$$\forall x \forall y \exists u \forall z (zEu \leftrightarrow (zEx \vee zEy)).$$

The intended meaning of zEu is $z \in u$. Graphically the axiom states that given x and y and a choice u for their union, the three constellations in Figure 1 are forbidden. To forbid the situations in (a) and (b), relation algebra suffices. (a) and (b) are expressed by **not** $x[E^{-1}{\circ}{-E}]u$ and **not** $y[E^{-1}{\circ}{-E}]u$. Using the dynamic implication $\backslash$, we can equivalently express this by $x[E\backslash E]u$ and $y[E\backslash E]u$.

To forbid the situation in (c) we need a name for the point x. If we assume that the only pair in the relation x is (x, x), then we can express it as **not** $u[E^{-1}{\circ}(-E \cap (-E{\circ}\mathsf{x}{\circ}\top))]y$ or equivalently, and much nicer, as

$$u[E\backslash(E \cup E{\circ}\mathsf{x}{\circ}\top)]y.$$

(Note that for x denoting a singleton relation $\{(x, x)\}$, $(-E \circ \mathsf{x} \circ \top) = -(E \circ \mathsf{x} \circ \top)$ is valid.) Putting this together, the union axiom is expressed by the equality

$$\downarrow^0_{\mathsf{x}}.((E\backslash E){\circ}((E\backslash(E \cup E{\circ}\mathsf{x}{\circ}\top)) \cap (E\backslash E)^{-1})) = \top. \tag{6}$$

Example 3 Already in 1915, A. Korselt showed that it is not possible to express by means of a relation algebraic (in)equality that the domain has at least four elements. In the expanded formalism it can be done as in (7).

$$\downarrow^0_{\mathsf{x}}.(\top\backslash[(0' \cap \top{\circ}\mathsf{x}{\circ}0'){\circ}0']) = \top. \tag{7}$$

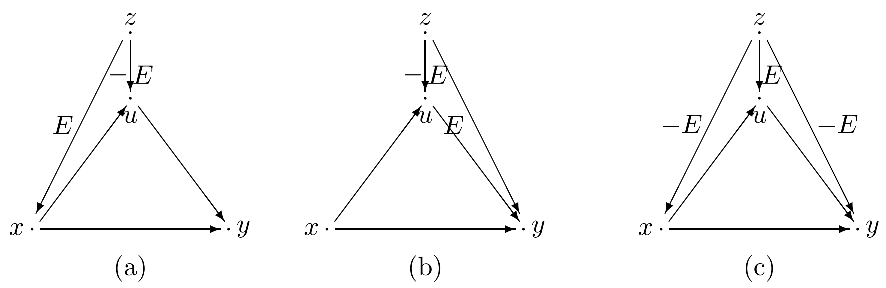

Figure 1: Situations forbidden by the union axiom

(7) is a close translation of the equivalent first-order statement $\forall x\forall y\forall z\exists u(x \neq u \wedge y \neq u \wedge z \neq u)$. Tarski improved on Korselt's example with the following sentence (Tarski was not that impressed by Korselt's sentence because it is a sentence which is, in his words, almost always true):

$$\forall x\forall y\forall z\exists u(xRu \wedge yRu \wedge zRu). \tag{8}$$

This is neatly expressed by

$$\downarrow^{0}_{\mathsf{x}}.(\top\backslash[(R\cap \top\circ\mathsf{x}\circ R)\circ R^{-1}]) = \top. \tag{9}$$

Example 4 Define the following "sibling" relation: two (different) persons are siblings if they share two different parents. In first-order logic, with xPy denoting that x is a parent of y:

$$x\mathsf{Sibling}y \iff x \neq y \wedge \exists p_1\exists p_2(p_1 \neq p_2 \wedge p_1Px \wedge p_2Px \wedge p_1Py \wedge p_2Py).$$

The "dynamic" definition in relation algebraic terms almost shows the two parents in action producing their offspring:[2]

$$\mathsf{Sibling} = 0'\cap \downarrow^{1}_{\mathsf{x}}.[(P\cap[(0'\cap P\circ\mathsf{x}\circ P^{-1})\circ P])^{-1}\circ\top].$$

One might want to add that the set of siblings and the set of parents are disjoint as well. This is obtained by replacing P everywhere by $(P\cap 0')$ and the last $\top$ by $0'$.

4 Full First-Order Expressivity

In this section, it is established that $RL{\downarrow}$ and first-order logic are equally strong in expressive power. Obviously the signature of first-order logic has to be restricted to just binary relation symbols. Call a closed $RL{\downarrow}$ term R and a first-order sentence φ equivalent if for all models $\mathfrak{M}$, φ is true in $\mathfrak{M}$ if and only if $\mathfrak{M} \models R = \top$. We

[2]One of the referees suggested the following translation in which x is a variable which is bound to a *pair* of elements, and $\downarrow_{\mathsf{x}}$ the corresponding binder.

$$\mathsf{Sibling} = 0'\cap \downarrow_{\mathsf{x}}.[P^{-1}\circ(P\cap(0'\circ P\circ\mathsf{x}\circ P^{-1}))].$$

Note that this is precisely the kind of variable ("they") used in the definition in English.

say that the two formalisms are equally strong in expressive power if for any relation algebraic term R there exists an equivalent first-order sentence φ and the other way round. Actually an even stronger form of equivalence holds. There exist recursive translations between the languages providing the equivalent formulation in the other language. Moreover, these translations preserve the atomic relation symbols.

The easy direction can be shown by a translation from $RL{\downarrow}$ terms to first-order formulas. This translation just copies the meaning definition of the $RL{\downarrow}$ terms. Let x, y, z be first-order variables different from the set of $RL{\downarrow}$ variables VAR. Define recursively the translation $ST_{(x,y)}(\cdot)$ from $RL{\downarrow}$ terms to first-order formulas:

$$\begin{array}{lcll}
ST_{(x,y)}(R) & := & xRy & \text{for all atomic } R\\
ST_{(x,y)}(1') & := & x{=}y & \\
ST_{(x,y)}(\mathsf{v}) & := & v{=}x \wedge x{=}y & \text{for } \mathsf{v} \text{ a variable}\\
\multicolumn{4}{l}{ST_{(x,y)}(\cdot) \text{ commutes with the booleans}}\\
ST_{(x,y)}(R^{-1}) & := & ST_{(y,x)}(R) & \\
ST_{(x,y)}(R{\circ}S) & := & \exists z(ST_{(x,z)}(R) \wedge ST_{(z,y)}(S)) & \\
ST_{(x,y)}(\downarrow^0_v.R) & := & \exists v(v = x \wedge ST_{(x,y)}(R)). &
\end{array}$$

(Note: in order for this translation to work, the variable z used for translating compositions of relations is not occurring free. Obviously this can be taken care of, e.g., by taking a fresh variable every time.) The proof of the following proposition is a straightforward induction on terms.

Proposition 2 For every model $\mathfrak{M}$, for all elements a, b in its domain, and for every closed $RL{\downarrow}$ term R it holds that

$$\begin{array}{rcl}
\mathfrak{M} \models a[R]b & \iff & \mathfrak{M} \models ST_{(x,y)}(R)\ [x \mapsto a, y \mapsto b]\\
\mathfrak{M} \models R = \top & \iff & \mathfrak{M} \models \forall x \forall y ST_{(x,y)}(R).
\end{array}$$

For the translation in the other direction, the first-order assignments to variables are separated into two parts: a part which is the denotation of a binary relation in (at most) free variables v_0 and v_1, and another part which handles the assignment of values to all other variables. This is because every relation algebraic term denotes a binary relation, and hence can be thought of as a formula in at most two free variables. This inspires a translation $(\cdot)^t$ from first-order formulas φ to $RL{\downarrow}$ terms which satisfies (10) below. For g an assignment to the variables $v_0, v_1, v_2, \ldots$, let $g_{\restriction 3}$ denote the restriction of g to the variables different from v_0 and v_1.

$$\mathfrak{M} \models \varphi\ [g] \iff \mathfrak{M} \models g(v_0)[\varphi^t]_{g_{\restriction 3}} g(v_1). \tag{10}$$

The translation $(\cdot)^t$ handles the variables v_0 and v_1 in a special way. For a smooth presentation, assume without loss of generality that v_0 and v_1 do not occur as arguments of relation symbols in the first-order formula (using two fresh variables this can always be achieved: e.g., $v_0Rv_3 \equiv \exists v_2(v_0 = v_2 \wedge v_2Rv_3)$). The translation is defined as follows: For atomic formulas,

$$(v_iRv_j)^t := @^0_{v_i} @^1_{v_j} R.$$

For equalities, there is a case distinction:

$$(v_0{=}v_1)^t := (v_1{=}v_0)^t := 1' \text{ and } (v_0{=}v_0)^t := (v_1{=}v_1)^t := \top.$$

8

For $i \notin \{0,1\}$,

$$\begin{array}{lclcl}(v_0 = v_i)^t & := & (v_i = v_0)^t & := & v_i \circ \top \\ (v_1 = v_i)^t & := & (v_i = v_1)^t & := & \top \circ v_i.\end{array}$$

For $i, j \notin \{0,1\}$,

$$(v_i{=}v_j)^t := (@^0_{v_i} v_j) \circ \top.$$

$(\cdot)^t$ commutes of course with the booleans. For the quantifiers, there is again a separation in cases:

$$(\exists v_0\varphi)^t := \top{\circ}\varphi^t \text{ and } (\exists v_1\varphi)^t := \varphi^t{\circ}\top,$$

and for $i \notin \{0,1\}$,

$$(\exists v_i\varphi)^t := \ \downarrow^0_x.(\top\circ \downarrow^0_{v_i}.@^0_x\varphi^t),$$

in which x is a variable which does not occur in φ.

The last clause of the translation describes dynamically what quantification is: store the current point of evaluation in an unoccupied memory place x, then change the point of evaluation at random, then store the value of that new point in v_i, and finally evaluate the formula with this new value for v_i at the original point stored in x. A straightforward induction proves

Proposition 3 For every model $\mathfrak{M}$, for every assignment g, and for every first-order formula φ, it holds that

$$\mathfrak{M} \models \varphi\ [g] \quad\iff\quad \mathfrak{M} \models g(v_0)[\varphi^t]_{g\restriction 3}\, g(v_1).$$

For sentences φ, $\mathfrak{M} \models \varphi$ iff $\mathfrak{M} \models \varphi^t = \top$.

5 Interpolation and Definability

An immediate consequence of the full first-order expressivity of *RL*↓ by language preserving translations is that *RL*↓ has an interpolation theorem, that is,

Theorem 4 For all closed *RL*↓ terms R, S, the following are equivalent:

1. $R{=}\top \Rightarrow S{=}\top$ is valid.
2. There exists an *RL*↓ term I constructed only from atomic symbols occurring both in R and in S for which $R{=}\top \Rightarrow I{=}\top$ and $I{=}\top \Rightarrow S{=}\top$ are both valid.

The interpolation theorem stated in this way fails for both RRA and RA [4]. One can similarly formulate a Beth definability theorem for RRA and RA (restricted to implicitly defined binary and unary relations only). For both these classes this fails as well [9]. Obviously, the thus defined Beth definability theorem holds for *RL*↓.

For completeness, a quick simple counterexample to interpolation for RRA is provided together with the required interpolant in *RL*↓. Let $p?$ abbreviate $(P \cap 1')$ and similarly for $q?$. Let A be the statement (7) in which the role of the variable x is now played by $p?$:

$$\top\backslash[(0' \cap \top{\circ}p?{\circ}0'){\circ}0'] = \top.$$

If A is true in a model, the model has at least four elements. Let C_1 be the term $\top\circ[q?\cap -(0'\circ q?\circ 0')]\circ\top$. $C_1 = \top$ expresses that there exists precisely one element which stands with itself in the q relation. Let C_2 be $\top\circ[0' \cap (-q)?\circ 0'\circ(-q)?\circ 0'\circ(-q)?]\circ\top$. The equality $C_2 = \top$ expresses that there exist at least three different elements which do not stand with themselves in the q relation.

It is thus clear that $A{=}\top \Rightarrow (C_1 \cap -C_2){=}\top$ is valid. Any interpolant should be written in the empty language and should express that there exist at least four elements. But in the empty language we can not express this, as observed above. In $RL{\downarrow}$, the interpolant is expressible by the equation (7).

Hajnal Andréka observed that such counterexamples can be constructed for any finite number $n > 3$. Thus also for obtaining the interpolation theorem an infinite similarity type seems to be needed. This even holds for much weaker classes of relational type algebras than RA cf., [8].

6 Conclusions and Further Research

Though equally expressive, $RL{\downarrow}$ and the full first-order language differ enormously. Expressing properties which are familiar in first-order logic (like the union axiom) in $RL{\downarrow}$ in an appealing way can be quite hard in the beginning. The sibling example shows that the same relation can have wildly different definitions. The first-order definition somehow forces the reader to consider the submodel consisting of the two siblings and two parents at once. The $RL{\downarrow}$ definition forces the reader to build this submodel during evaluation of the term. Such different modes of evaluation suggest possibly also different algorithms for doing model-checking and theorem proving.

Axiomatics were not discussed. I conjecture that complete axiom systems can rather easily be defined using irreflexivity style rules or something like the Paste rule from [2].

A related research topic is the relation with finite variable fragments of first-order logic and with the proof theoretic approximations to RRA studied by Hirsch, Hodkinson and Maddux [6]. Many questions can be asked here. We mention a few. Is it possible to equationally define a class of $RL{\downarrow}$ algebras whose relation algebra reducts generated the variety RA_n (the class of relation algebras with an n-dimensional relational basis)? How many variables are needed? Is $RL{\downarrow}$ restricted to n variables equally expressive as first-order logic with n variables?

Acknowledgments

The author is supported by NWO grant 612-062-001. Part of this research was carried out during a visit to Patrick Blackburn in Saarbrücken which has been made possible by a grant for international collaboration with the DFKI and the University of Saarbrücken, awarded by the Dutch Research Council NWO to the Dutch research School in Logic. Thanks are due to Jorge Petrucio Viana and Renata Pereira de Freitas and to the anonymous referees for valuable comments.

References

[1] C. Areces, P. Blackburn, and M. Marx. Hybrid logics. Characterization, interpolation and complexity. Technical Report PP-1999-07, Institute for Logic, Language and Computation, University of Amsterdam, 1999. To appear in Journal of Symbolic Logic.

[2] P. Blackburn. Internalizing labelled deduction. *Journal of Logic and Computation*, 10:136–168, 2000.

[3] P. Blackburn. Representation, reasoning, and relational structures: a hybrid logic manifesto. *Logic Journal of the IGPL*, 8(3):339–365, 2000.

[4] S. Comer. Classes without the amalgamation property. *Pacific Journal of Mathematics*, 28:309–318, 1969.

[5] M. Frías, G. Baum, and A. Haeberer. Fork algebras in algebra, logic and computer science. *Fundamenta Informaticae*, 32:1–25, 1997.

[6] R. Hirsch and I. Hodkinson. Relation algebras with n-dimensional relational bases. *Annals of Pure and Applied Logic*, 101:227–274, 2000.

[7] A. Kwatinetz. *Problems of expresibility in finite languages*. PhD thesis, University of California, Berkeley, 1981.

[8] M. Marx. Amalgamation in relation algebras. *Journal of Symbolic Logic*, 63(2):479–484, 1998.

[9] I. Sain. Beth's and Craig's properties via epimorphisms and amalgamation in algebraic logic. In C. H. Bergman, R. D. Maddux, and D. L. Pigozzi, editors, *Algebraic logic and universal algebra in computer science*, volume 425 of *Lecture Notes in Computer Science*, pages 209–226. Springer-Verlag, Berlin, 1990.

[10] A. Tarski. On the calculus of relations. *Journal of Symbolic Logic*, 6:73–89, 1941.

[11] A. Tarski and S. Givant. *A Formalization of Set Theory without Variables*, volume 41. AMS Colloquium publications, Providence, Rhode Island, 1987.

Modal Logic with Bounded Quantification over Worlds

Rogier M. van Eijk[1] Frank S. de Boer[1]
Wiebe van der Hoek[1,2,3]
John-Jules Ch. Meyer[2]

[1]Institute of Information and Computing Sciences, Utrecht University
P.O. Box 80.089, 3508 TB Utrecht, The Netherlands
Email: {rogier,frankb,wiebe,jj}@cs.uu.nl
[2]Department of Philosophy, Utrecht University
The Netherlands
[3]Department of Computer Science, University of Liverpool
United Kingdom

Abstract

In this paper[1], we present a logical framework that combines modality with a first-order variable-binding mechanism. The logic, which belongs to the family of hybrid languages, differs from standard first-order modal logics in that quantification is not performed inside the worlds of a model, but the worlds in the model themselves constitute the domain of quantification. The locality principle of modal logic is preserved via the condition that in each world, the domain of quantification is given by a subset of the entire set of worlds in the model. In comparison with standard hybrid languages, the logic covers separate mechanisms for navigation and for variable-binding and formalizes reasoning about the worlds of a model in terms of equational logic. We show that the logic is semantically characterized by a generalization of classical bisimulation, called history-based bisimulation, and study the application of the logic to describe and reason about network topologies.

Keywords: modal logics, equational logic, bounded quantification over worlds, history-based bisimulation, network topologies, hybrid languages.

1 Introduction

In order to increase the expressiveness of modal logics, during the last decades, a new family of logics has been introduced that combine modal operators with first-order variable-binding mechanisms. Characteristic of these logics, which are referred to as the family of *hybrid languages* [4], is that quantification is not performed inside

[1]This paper is a revised version of [6].

the worlds of a model like in standard first order modal logic [7], but instead, the worlds themselves constitute the domain of quantification. In particular, standard hybrid languages, which have originally been developed to increase the expressiveness of tense logics [5], extend modal logic with a collection of *nominals* that are used to label the worlds of a model. These nominals are propositional formulas that are true at exactly one world, and as such are employed as global, unique names for worlds. Further extensions cover operators to quantify over the worlds in a model, to bind variables to the current world, to jump to worlds denoted by a particular nominal or variable, and operators that combine modal, binding and jumping aspects. In this paper, we present a logical framework that belongs to this family of hybrid languages, which, due to its different starting-point and underpinning motivations provides a new perspective on hybrid languages.

The starting point of the framework is an explicit separation between the mechanisms of navigation and variable-binding. That is, in a particular world of a model, we distinguish between the worlds that are directly *accessible* from it and the worlds over which can be *quantified*. This starting point yields a general framework that can be instantiated in different ways. The framework for instance allows a global domain of quantification, but in particular, also *bounded* forms of quantification where in each world of a model, there is a restricted domain over which is quantified, like for instance the set of worlds that are directly accessible or the worlds that are accessible by following a finite number of successive links. Additionally, in comparison with standard hybrid languages, the logic is tailored to reason about the worlds of a model in terms of *equational logic*.

The paper is organized as follows. In Section 2, we motivate our research in extensions of modal logics by considering network topologies and argue that existing logics such as basic modal logic and graded modal logic are not suited to reason about network topologies. In Section 3, we present the syntax and semantics of our logical framework. Subsequently, in Section 4, we establish a semantic characterization of the logic, which is based on a generalization of the classical notion of bisimulation equivalence. Instead of relating worlds, this new type of bisimulation relates tuples that are comprised of a world together with a sequence of worlds. These additional sequences are employed to represent variable bindings that are generated during the evaluation of formulas. In Section 5, we consider the relation of the framework with the more standard hybrid languages and describe several interesting extensions of the framework that form the subject of future research.

2 Network Topologies

An important application of the logical framework presented in this paper is the description of network topologies. Formally, such a network topology is represented by a directed graph whose nodes denote the agents in the system and whose edges make up the accessibility relation, describing what agents know about each other.

Definition 1 A *network topology* is a tuple of the form:

$$\mathcal{N} = \langle W, R \rangle,$$

where W is a set of agents and $R \subseteq W \times W$ denotes the *accessibility* relation on W. We use the notation $R(w)$ to denote the set $\{u \in W \mid R(w,u)\}$ of agents that are known by the agent w.

Seen from a logical point of view, these network topologies constitute Kripke models (without a valuation function) that are employed in the semantics of modal logic [11]. This observation naturally leads to a description of network topologies by means of modal logic.

The basic modal language is defined as follows.

Definition 2 Formulas φ in the *basic modal language* $\mathcal{L}_0$ are generated using the following BNF-grammar:

$$\varphi ::= \mathit{true} \;\mid\; \varphi_1 \wedge \varphi_2 \;\mid\; \neg\varphi \;\mid\; \Diamond\varphi.$$

Furthermore, we assume the usual abbreviations *false* for $\neg \mathit{true}$, $\varphi_1 \vee \varphi_2$ for $\neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\varphi_1 \rightarrow \varphi_2$ for $\neg\varphi_1 \vee \varphi_2$ and $\varphi_1 \leftrightarrow \varphi_2$ for $\varphi_1 \rightarrow \varphi_2 \wedge \varphi_2 \rightarrow \varphi_1$.

A modal formula is either equal to *true*, the conjunction of two modal formulas, the negation of a modal formula, or the operator $\Diamond$ applied to a modal formula. It is the operator $\Diamond$ that gives the language the modal flavor; it has various readings like for instance the interpretation of expressing *possibility*. The dual $\Box$ of this operator, which is defined as $\neg\Diamond\neg$, can be thought of as denoting *necessity*.

The interpretation of modal formulas is given in the following truth definition.

Definition 3 Given a network topology $\mathcal{N} = \langle W, R\rangle$, a world $w \in W$ and a formula $\varphi \in \mathcal{L}_0$, the *truth definition* $\mathcal{N}, w \models \varphi$ is given by:

$$\begin{array}{lcl} \mathcal{N}, w \models \mathit{true} & & \\ \mathcal{N}, w \models \varphi_1 \wedge \varphi_2 & \Leftrightarrow & \mathcal{N}, w \models \varphi_1 \text{ and } \mathcal{N}, w \models \varphi_2 \\ \mathcal{N}, w \models \neg\varphi & \Leftrightarrow & \mathcal{N}, w \not\models \varphi \\ \mathcal{N}, w \models \Diamond\varphi & \Leftrightarrow & \mathcal{N}, v \models \varphi \text{ for some } v \in R(w) \end{array}.$$

For instance, $\mathcal{N}, w \models \Box\Diamond\mathit{true}$ expresses that in the network topology $\mathcal{N}$, all acquaintances of the agent w are acquainted to an agent.

Figure 1: Two bisimilar network topologies

The language $\mathcal{L}_0$ is however not expressive enough to describe and reason about network topologies. Consider for instance the network topologies in Figure 1, which from a local perspective denote distinct situations. That is, a logic for network topologies should be able to distinguish between the situation that an agent's circle of acquaintances is comprised of two agents and the situation that this circle consists of only one agent. However, the basic language $\mathcal{L}_0$ lacks the expressive power to distinguish between both network topologies; i.e., there does not exist a formula that is true in the left network and not in the right one. Formally, this follows from the fact that these networks are bisimilar.

Extensions of the basic modal language that deal with numbers of successors are the *graded modal languages* [10]. Rather than one modal operator $\Diamond$ the graded language contains a set $\{\Diamond_n \mid n \geq 0\}$ of operators. A formula of the form $\Diamond_n \varphi$ expresses that there exist more than n accessible worlds in which φ holds. Hence, graded modal logic distinguishes between the network topologies in Figure 1. For instance, the formula $\Diamond_1 \mathit{true}$ is true for the agent in the left network but not for the agent in the right one.

Graded modal languages are still not suitable to describe network topologies. For instance, consider the two networks in Figure 2, which denote a loop and its infinite unfolding, respectively. The left network topology consists of an agent that knows only of itself, while in the second network there is an infinite chain of agents that know of each other. Since it can be shown that graded modal logic does not possess the expressive power to distinguish between these bisimilar networks, this logic is also not adequate to reason about network topologies.

3 Bounded Quantification over Worlds

Our analysis of the reason why basic modal logic and its extension with graded modalities are not fit to reason about network topologies, is that they lack a mechanism for dealing with *world identities*.

...

Figure 2: Loop and its infinite unfolding

For instance, if we would be able to compare the identities of the accessible worlds in Figure 1, and to compare the identity of the current world with that of its successor world in Figure 2, then we would be able to distinguish between these network topologies. This observation naturally leads to an extension of the basic modal logic with a collection of variables to denote the worlds of a model together with an operator to bind them.

Definition 4 Given a set *Var* of variables, terms t and formulas φ of the *extended modal language* $\mathcal{L}_1$ are generated using the following BNF-grammar:

$$\begin{array}{lcl} t & ::= & \mathit{self} \mid x \\ \varphi & ::= & (t_1 = t_2) \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \Diamond\varphi \mid \exists x(\varphi), \end{array}$$

where x ranges over the variables of *Var*.

We assume the usual abbreviation $\forall x \varphi$ for $\neg\exists x \neg\varphi$. The formula *true* can be represented by the formula $\mathit{self} = \mathit{self}$. A formula φ is called a *sentence* if it contains no free variables, i.e., all variables x in φ occur in the scope of a quantifier $\exists x$.

Terms of the language $\mathcal{L}_1$ are variables denoting worlds and a special constant *self* denoting the *current* world. An atomic formula is of the form $t_1 = t_2$, expressing that terms t_1 and t_2 denote the same world. We omit propositional variables here since their treatment is standard and orthogonal to the other logical operators. Additionally, the operator $\exists x$ binds the variable x to some world in the domain of quantification of the current world.

We define the following general models for the extended modal language.

Definition 5 A *(generalized) Kripke model* for $\mathcal{L}_1$ is a tuple of the form:

$$\mathcal{M} = \langle W, R, D \rangle,$$

where W is the set of worlds, $R \subseteq W \times W$ denotes the *accessibility* relation on W and $D \subseteq W \times W$ defines the domains of quantification. We write $D(w)$ to denote the domain of quantification $\{u \in W \mid D(w,u)\}$ of the world w.

In addition to an accessibility relation R, a model contains a relation D that defines for each world the set of worlds over which can be quantified. There are various possible instantiations of this domain relation D, like for instance $\{(w,w) \mid w \in W\}$, which only allows variables to be bound to the current world, and the universal relation $\{(w_1,w_2) \mid w_1, w_2 \in W\}$ which enables the binding of variables to any world in the model.

In particular, in the case of *network topologies*, we would like to be able to quantify over precisely the agents that are known. In other words, we assume that network topologies are Kripke models in which the domain relation coincides with the accessibility relation, that is, $R = D$.

Still, there are other possible instantiations. As a final example we mention a domain relation that is given by the transitive closure of the accessibility relation. In this case we are able to quantify over all worlds that are accessible by following one or more links.

Before we define the interpretation of $\mathcal{L}_1$ in terms of Kripke models, we introduce some helpful notation.

Definition 6 Given a partial function $f : X \to Y$, we use the notation $f(x) = \perp$ to denote that f is not defined for x. Additionally, the *domain* of f is defined by $dom(f) = \{x \mid f(x) \neq \perp\}$. Its *range* is given by $ran(f) = \{y \in Y \mid \text{ exists } x \in X \text{ with } f(x) = y\}$. Finally, we write $f[y/x]$ to denote the function that behaves like f except on the input x for which it yields the output y.

The interpretation of terms and formulas in $\mathcal{L}_1$ is given via the following truth definition, where we use the notion of an assignment for the interpretation of variables. Such an assignment is a function $s : Var \to W$ of finite domain, which maps variables to worlds in the set W.

Definition 7 Given a model $\mathcal{M} = \langle W, R, D \rangle$, the *interpretation* $[\![t]\!]_{w,s}$ of a term t in a world $w \in W$ under an assignment function $s : Var \to W$, is defined by:

$$\begin{array}{lcl} [\![self]\!]_{w,s} & = & w \\ [\![x]\!]_{w,s} & = & s(x) \end{array}$$

5

The *truth definition* $\mathcal{M}, w, s \models \varphi$ is given by:

$$\begin{array}{lcl}
\mathcal{M}, w, s \models (t_1 = t_2) & \Leftrightarrow & [\![t_1]\!]_{w,s} = [\![t_2]\!]_{w,s} \\
\mathcal{M}, w, s \models \varphi_1 \wedge \varphi_2 & \Leftrightarrow & \mathcal{M}, w, s \models \varphi_1 \text{ and } \mathcal{M}, w, s \models \varphi_2 \\
\mathcal{M}, w, s \models \neg\varphi & \Leftrightarrow & \mathcal{M}, w, s \not\models \varphi \\
\mathcal{M}, w, s \models \Diamond\varphi & \Leftrightarrow & \mathcal{M}, v, s \models \varphi \text{ for some } v \in R(w) \\
\mathcal{M}, w, s \models \exists x\varphi & \Leftrightarrow & \mathcal{M}, w, s[v/x] \models \varphi \text{ for some } v \in D(w)
\end{array}$$

Additionally, we have $\mathcal{M}, w \models \varphi$ if for all assignments s it holds that $\mathcal{M}, w, s \models \varphi$. Finally, we write $\mathcal{M} \models \varphi$ if $\mathcal{M}, w \models \varphi$ holds for all $w \in W$.

Note the difference in the truth definition between the operators $\Diamond$ and $\exists$ with respect to the point of evaluation: in the truth definition of the former operator there is a shift in perspective, viz. from w to v, whereas in the latter, the point of view w remains fixed. In other words, $\exists$ quantifies over the current domain while the operator $\Diamond$ is used to change the scope of quantification.

Finally, note that the constant *self* constitutes a *non-rigid designator* [7] in the sense that its denotation differs among the worlds in a model; in particular, in each world the denotation of this designator is the world itself.

The logic $\mathcal{L}_1$ distinguishes between the network topologies in Figures 1 and 2. For instance, the formula $\exists x \exists y \neg(x = y)$ is true in the left network in Figure 1 but not in the right one. Secondly, the formula $\exists x(x = \mathit{self})$ distinguishes between the two networks in Figure 2. An example of a distinguishing formula that does not contain the constant *self*, is the formula $\exists x \Diamond \exists y(y = x)$, which is true in the left network in Figure 2 but not in the right one.

The general set up of the framework allows us to study the connections between the accessibility relation and the domains of quantification that the language $\mathcal{L}_1$ can express. We consider the property that the domain relation is a contained in the accessibility relation.

Observation 8 For all models $\mathcal{M} = \langle W, R, D\rangle$ and worlds $w \in W$ the following holds:

$$\mathcal{M}, w \models \forall x \Diamond (x = \mathit{self}) \text{ iff } D(w) \subseteq R(w).$$

In Corollary 18 below, we prove that there does not exist a formula that expresses $R(w) \subseteq D(w)$, for all w. However, a straightforward refinement of the language is its extension with the inverse operator of $\Diamond$, denoted by $\Diamond^{-1}$, which has a natural interpretation in the context of network topologies: It denotes the *is-known-by* relation.

Definition 9 The interpretation of the inverse navigation operator $\Diamond^{-1}$, is defined by:

$$\mathcal{M}, w, s \models \Diamond^{-1}\varphi \Leftrightarrow \mathcal{M}, v, s \models \varphi \text{ for some } v \text{ with } w \in D(v).$$

With this extra operator, we obtain the following result.

Observation 10 Given a model $\mathcal{M} = \langle W, R, D\rangle$ with a reflexive domain relation D, for all worlds $w \in W$, we have $R(w) \subseteq D(w)$ if and only if:

$$\mathcal{M}, w \models \exists x(x = \mathit{self} \wedge \Box(\exists y(y = \mathit{self} \wedge \Diamond^{-1}(x = \mathit{self} \wedge \exists z(z = y))))).$$

6

As mentioned before, network topologies are Kripke models in which the domain relation coincides with the accessibility relation. Let us consider some properties of network topologies we can express using the language $\mathcal{L}_1$.

Example 11 (Network Topologies) First of all, the formula

$$\exists x \Diamond (x = \mathit{self})$$

can be thought of expressing "knowing yourself." Secondly, the formula

$$\exists x (x = \mathit{self} \land \Box \Diamond x = \mathit{self})$$

is true in a world in case all accessible worlds have in turn access to this world. In other words, it expresses "everyone that I know, knows me." Additionally, the formula

$$\exists xy(\neg(x = y) \land \Diamond(x = \mathit{self} \land \neg\Diamond y = \mathit{self}) \land \Diamond(y = \mathit{self} \land \neg\Diamond x = \mathit{self}))$$

is true in a particular world, in case there are two distinct accessible worlds that are not accessible to one another. Informally, it can be thought of as expressing "I know two agents that do not know each other."

Finally, we illustrate that quantification does not commute with modality. Consider the formula

$$\exists x \Box (x = \mathit{self}),$$

which is true in a world in case there is exactly one accessible world, and which can be thought of expressing "I know of exactly one agent." On the other hand, the formula

$$\Box \exists x (x = \mathit{self})$$

expresses something different, namely that "everyone that I know, knows itself."

In the remainder of this section, we consider the finite model property and the decidability of the logic.

Lemma 12 The language $\mathcal{L}_1$ does not satisfy the finite model property.

Proof We show that $\mathcal{L}_1$ can enforce infinite domains. Let $P(x, y)$ stand for the following formula:

$$\Diamond(x = \mathit{self} \land \Diamond y = \mathit{self}),$$

which expresses that the world x is accessible from the current world, and from x the world y is accessible. Let φ denote the conjunction of the following formulas $\exists x(\mathit{true})$, which reflects a nonempty domain, $\forall x(\neg P(x, x))$ expressing the irreflexivity of the relation P, the formula $\forall x \forall y \forall z((P(x, y) \land P(y, z)) \to P(x, z))$ denoting transitivity, and $\forall x \exists y(P(x, y))$ expressing seriality. It is not difficult to see that if this formula is true in a particular world then its domain of quantification must be infinite (see Figure 3).

. . .

Figure 3: Construction of an infinite domain

In the above proof, the use of the constant *self* is not essential. The language $\mathcal{L}_1$ without this constant does not satisfy the finite model property either. This can be shown in a similar manner using the following definition of the formula $P(x, y)$:

$$\Diamond(\exists u(u = x) \wedge \Diamond\exists u(u = y)).$$

Thus, $P(x, y)$ expresses that y is in the domain of a world that can be accessed from some accessible world (with respect to the current world) that has x in its domain.

The crux in the proof of Lemma 12 is the construction of an infinite *domain of quantification*. It is still an open issue for future research whether $\mathcal{L}_1$ satisfies the finite model property in case we restrict to Kripke models in which the domains of quantification are finite.

In [2], it is shown that the hybrid language consisting of the basic modal language extended with variables and an operator $\downarrow x$ to bind the variable x to the current world, has an undecidable validity problem. Since a formula $\downarrow x(\varphi)$ can be modeled by $\exists x(x = \mathit{self} \wedge \varphi)$ in our language, we derive that the language $\mathcal{L}_1$ is undecidable. This also implies that the language $\mathcal{L}_1$ is not part of the *guarded fragment* of first-order logic [1], since this fragment has a decidable validity problem.

Moreover, an interesting question arises with respect to the role of the constant *self* in this result. The language $\mathcal{L}_1$ without this constant *self* is also not part of the guarded fragment of first-order logic. Yet it is an open issue for future research whether the validity problem of this sublanguage of $\mathcal{L}_1$ is decidable.

4 Semantic Characterization

In this section, we study the expressiveness of the language $\mathcal{L}_1$. In particular, we address the issue of what properties the language can express and what properties are beyond its expressive power. The central result is a *semantic characterization* of the language, which defines the conditions under which two Kripke models satisfy precisely the same formulas of $\mathcal{L}_1$.

For the basic modal language $\mathcal{L}_0$ the semantic characterization is given by the notion of a *bisimulation* [3, 9]. That is, two models satisfy the same basic modal formulas if and only if they are bisimilar. In this paper, we introduce a new notion of bisimulation, called *history-based bisimulation*, which extends classical bisimulation with a mechanism to handle quantifications. Instead of relating worlds, this new type of bisimulation relates tuples that are comprised of a world together with a injective sequence of worlds. These additional sequences are employed to represent variable bindings that are generated during the evaluation of formulas.

Let us first introduce some helpful notation with respect to injective sequences.

Definition 13 Given a set of worlds W an *injective sequence* over W, or *sequence* for short, is a function $\bar{v} : I\!N \rightarrow W$ of finite domain, which satisfies for all $i, j \in dom(\bar{v})$:

$$\bar{v}(i) = \bar{v}(j) \ \Rightarrow \ i = j.$$

Additionally, ϵ denotes the empty sequence; that is, $\epsilon(i) = \bot$ for all $i \in I\!N$.

Finally, we define:

$$\bar{v} \bullet w = \begin{cases} \bar{v}[w/i] & \text{if } w \notin ran(\bar{v}) \\ \bar{v} & \text{otherwise,} \end{cases}$$

where $i \in I\!N$ is the next index not part of $dom(\bar{v})$. Thus, $\bar{v} \bullet w$ denotes the extension of the sequence $\bar{v}$ with w in case w does not already occur in $\bar{v}$, and denotes $\bar{v}$ itself, otherwise.

An injective sequences $\bar{v} : I\!N \rightarrow W$ is an *abstraction* of an assignment $s : Var \rightarrow W$ that just contains the information that is needed in the semantic characterization. That is, an assignment s is represented by a sequence that consists of the elements in the range of s in some particular order. This representation thus abstracts from any repetitions of worlds and the particular variable names. From a *computational* point of view, the advantage of sequences in comparison with assignments is that they give rise to a *decidable* semantic characterization (see Observation 15).

Next, we introduce the notion of a history-based bisimulation.

Definition 14 Given the models $\mathcal{M}_1 = \langle W_1, R_1, D_1 \rangle$ and $\mathcal{M}_2 = \langle W_2, R_2, D_2 \rangle$, a non-empty relation $\sim$ is a *history-based bisimulation*, if $(w_1, \bar{v_1}) \sim (w_2, \bar{v_2})$ implies the following:

(self) $w_1 = \bar{v_1}(i)$ iff $w_2 = \bar{v_2}(i)$, for all $i \in I\!N$,
(bisim) if $u_1 \in R_1(w_1)$ then there exists $u_2 \in R_2(w_2)$ with $(u_1, \bar{v_1}) \sim (u_2, \bar{v_2})$,
(var) if $u_1 \in D_1(w_1)$ then there exists $u_2 \in D_2(w_2)$ with $(w_1, \bar{v_1} \bullet u_1) \sim (w_2, \bar{v_2} \bullet u_2)$

and conditions similar to **(bisim)** and **(var)** from $\mathcal{M}_2$ to $\mathcal{M}_1$.

Additionally, we define $w_1 \sim w_2$ to hold in case $(w_1, \epsilon) \sim (w_2, \epsilon)$.

The reason why the notion of bisimulation is called *history-based* is that the injective sequences record the worlds in the domains of the encountered worlds that have been bound by a particular variable. We could say that they make up a history of landmarks: if during navigation we arrive at a world that has such a landmark in its domain, then in the other model, we should be at a world that has the corresponding landmark in its domain.

It is worth remarking here that the notion of a history-based bisimulation is quite different from the notion of a *history-preserving bisimulation* [8]. The latter is a very strong notion saying that two worlds are history-preserving bisimilar in case they are related by a bisimulation and additionally, the respective submodels consisting of the worlds that can reach the world via the accessibility relation, are isomorphic.

If we restrict ourselves to models with a finite number of worlds, the notion of a history-based bisimulation is decidable.

Observation 15 Given models $\mathcal{M}_1$ and $\mathcal{M}_2$ with a *finite* number of worlds, for all worlds w_1 in $\mathcal{M}_1$ and w_2 in $\mathcal{M}_2$, it is *decidable* whether there exists a history-based bisimulation $\sim$ such that $w_1 \sim w_2$.

Note that it is crucial here that injective sequences do not contain repetitions of worlds. This implies that there exists a bound on the number of applications of rule **(var)** that we need to consider.

Before we phrase the semantic characterization of the language $\mathcal{L}_1$ in Theorem 17, we define the notion of an image-finite world.

Definition 16 Given a model $\mathcal{M} = \langle W, R, D\rangle$, a world $w \in W$ is called *image-finite* if $R(v)$ and $D(v)$ are finite for all v with $(w, v) \in R^*$, where R^* denotes the reflexive, transitive closure of R.

Properly, we do not need the assumption of image-finiteness, as analogous to the proof of the semantic characterization of standard modal logic, we could use ultrafilter extensions [3]. However, for the sake of simplicity we adopt this property here.

Theorem 17 (Semantic characterisation) Given two models $\mathcal{M}_1$ and $\mathcal{M}_2$, for all worlds w_1 from $\mathcal{M}_1$ and w_2 from $\mathcal{M}_2$ the following hold:

(*i*) if $w_1 \sim w_2$ for some history-based bisimulation $\sim$ then for all *sentences* $\varphi \in \mathcal{L}_1$ we have $\mathcal{M}_1, w_1 \models \varphi \Leftrightarrow \mathcal{M}_2, w_2 \models \varphi$

(*ii*) if w_1 and w_2 are image-finite and $\mathcal{M}_1, w_1 \models \varphi \Leftrightarrow \mathcal{M}_2, w_2 \models \varphi$ for all *sentences* $\varphi \in \mathcal{L}_1$, then $w_1 \sim w_2$ for some history-based bisimulation $\sim$.

The proof of the semantic characterization is given in the appendix. Let us consider some applications of the result.

Figure 4: A confluent and non-confluent model

First of all, consider the confluent model and the non-confluent model in Figure 4. In case their domain relations are equal to the accessibility relations, these two models are related by a history-based bisimulation, implying that the language $\mathcal{L}_1$ cannot distinguish between them. However, note that in case their domain relations are equal to the *transitive closure* of the accessibility relations, the formula $\exists x \Box\Box(x = \mathit{self})$, which is true in the left network but not in the right one, is an example of a distinguishing formula.

Secondly, the language $\mathcal{L}_1$ cannot express the property that the accessibility relation is contained in the domain relation, as stated in the following result.

Corollary 18 There does not exist a formula $\varphi \in \mathcal{L}_1$ such that for all models $\mathcal{M} = \langle W, R, D\rangle$ and worlds $w \in W$ we have: $\mathcal{M}, w \models \varphi$ iff $R(w) \subseteq D(w)$.

Proof Consider the model $\mathcal{M} = \langle W, R, D\rangle$ and a state $w \in W$ such that

- W is an *infinite* set of worlds,
- R satisfies $(w, v) \in R$ for all $v \in W$,
- D satisfies $(w, v) \in D$ for all $v \in W$.

Additionally, we have a model $\mathcal{M}^\star$ that extends $\mathcal{M}$ with a world $\star$ defined by:

$$\mathcal{M}^\star = \langle W \cup \{\star\}, R \cup \{(w, \star)\}, D\rangle.$$

These two models are related by the following history-based bisimulation $\sim$. For all $u \in W$, sequences $\bar{v}$ over W and $u' \in W \setminus ran(\bar{v})$:

$$\begin{array}{rcl} (u, \bar{v}) & \sim & (u, \bar{v}) \\ (u', \bar{v}) & \sim & (\star, \bar{v}) \end{array}$$

Note that such a world u' exists since $ran(\bar{v})$ is finite while the set W is infinite. Consequently, by Theorem 17 we obtain that the language $\mathcal{L}_1$ cannot distinguish between these two models, and as $R(w) \subseteq D(w)$ and $R(w) \cup \{(w, \star)\} \not\subseteq D(w)$, we derive the desired result.

5 Related Work and Future Research

Our framework is closely connected to the work on *hybrid languages*, which also combine modality with first-order variable-binding mechanisms [4]. In particular, hybrid languages extend the basic modal language $\mathcal{L}_0$ with a collection of *nominals* that are used to label worlds in a model. These nominals are propositional formulas that are true at exactly one world in a model, and so to speak are employed as global *unique* names for worlds. Further extensions additionally incorporate operators of the form $@_t$ to *jump* to the world that is denoted by the term t, as well as operators to *bind* variables; e.g., the operator $\downarrow x$ to bind the variable x to the current world and the existential quantifier, which we denote as $\bar{\exists} x$ to distinguish it from the quantifier $\exists x$ from $\mathcal{L}_1$, which quantifies over all worlds of a model.

First of all, the operator $\downarrow x$ to bind the variable x to the current world can be represented in the language $\mathcal{L}_1$ as follows:

$$\downarrow x(\varphi) \ \Leftrightarrow\ \exists x(x = \mathit{self} \wedge \varphi).$$

The operator corresponds to existential quantification in the class of models in which for each world the domain of quantification consists of only the world itself; that is, in the class:

$$\{\mathcal{M} \mid \mathcal{M} \models \exists x(x = \mathit{self} \wedge \forall y(y = x))\}.$$

Additionally, the hybrid quantifier $\bar{\exists} x$ ranges over the *entire* set of worlds in a model. In our framework this operator corresponds with existential quantification in the class of models in which the domain of quantification of each world coincides with the entire set of worlds.

Finally, we mention the hybrid operator $@_t$ that is used to jump to the world denoted by the term t. The truth definition of this operator can be given as follows:

$$\mathcal{M}, w, s \models @_t\varphi \ \Leftrightarrow\ \mathcal{M}, v, s \models \varphi, \text{ where } v = [\![t]\!]_{w,s}.$$

This operator has no counterpart in our framework due to the fact that in each world, it allows moving to worlds that are not necessarily reachable via the accessibility relation. This is in contrast with one of our underlying assumptions that in a world one cannot move to arbitrary worlds but only to those worlds which are accessible.

Let us consider the major differences between our framework and the hybrid languages as described above. First of all, an important characteristic of the hybrid languages is the treatment of terms as formulas. That is, analogous to formulas, a term can be true or false at a world of a model: It is true if its denotation is exactly the current world and is false otherwise. In contrast, our logic is based on a more conventional ontology which distinguishes between terms and formulas; i.e., a term denotes a world and a formula a Boolean value. Consequently, our framework formalizes reasoning about the identities of the worlds of a model directly in terms of *equational logic*.

A second difference with the hybrid languages, is our separation of navigation and variable-binding mechanisms; that is, in our framework, there is one operation for navigating a model and another operation for bounded quantification over worlds. This treatment allows us to study these different mechanisms in isolation as well as to examine their interactions. In contrast, hybrid languages cover operators, such as the operators $\Downarrow x$ and Σx in [4], that embody both navigation and binding aspects.

Thirdly, in the hybrid languages, the interpretation of nominals is *absolute*, which means that each of these constants denotes a unique world in the model. In contrast, our framework allows natural extensions with *relative* constants, which are constants whose interpretation depends on the current world. Such an extension can be used when reasoning about the ambiguities of *names* in, for example, multi-agent topologies; that is, situations in which one agent is known by other agents under different names. Formally, we may extend our language $\mathcal{L}_1$ with a countable set C of names, with typical element c. A term t in the extended language, which is called $\mathcal{L}_2$, is thus either a variable x, the constant *self*, or a name $c \in C$. Formulas are defined as in Definition 4 and they are interpreted over the following models:

$$\langle W, R, D, I \rangle,$$

where W is a set of worlds, $R \subseteq W \times W$ denotes the accessibility relation, and I is a total function which assigns to each $w \in W$ an interpretation $I(w)$ of each name $c \in C$, that is, $I(w) \in C \rightarrow W$. Furthermore, $D = \{\langle w, I(w)(c) \rangle \mid c \in C\}$. In other words, for each world the domain of quantification is given by the local denotations of the constants.

The definition of the truth of a formula φ in the extended language $\mathcal{L}_2$ involves a straightforward adaptation of the truth definition of the language $\mathcal{L}_1$ and is therefore omitted. Instead, we explain here the use of quantification in the description of the ambiguities to which names may give rise. First, we observe that without quantification we cannot describe phenomena like that one agent is known by different agents under different names. For example, given an agent w, we cannot describe the situation that $I(w)(c) = I(w')(c)$, for some $(w, w') \in R$, simply because the modal operators induce a "context switch," that is, a different interpretation of the names. However this situation can be described using quantifiers simply by the formula:

$$\exists x(x = c \wedge \Diamond(x = c)).$$

So, we bind the value of the constant c to the variable x, and use the fact that the interpretation of the variables is fixed, that is, does not change when "moving" from one agent to another. This example illustrates the difference with the variable-binding mechanism of *first-order logic*: in first-order logic, the formula $\exists x(x = t \wedge \varphi)$ can be modeled by the substitution $\varphi[t/x]$ of t for x in φ.

Another example of a further extension of our framework concerns a formalization of reasoning about the identities of *objects* as they appear during the execution of an object-oriented program. A constant $c \in C$ is interpreted in this application as a *pointer attribute*. Object structures can be modeled as Kripke models that contain a family of deterministic accessibility relations; one for each pointer attribute.

Definition 19 A *deterministic (generalized) Kripke model* is a pair (W, I), where as above, W is a set of worlds and $I \in W \to (C \to W)$.

The set W represents the set of existing objects and I describes the pointer structure. These models support the following natural multi-modal extension $\mathcal{L}_3$ of our basic logic, where each pointer attribute is also used as a *modal operator*:

$$\varphi \quad ::= \quad (t_1 = t_2) \;\mid\; \neg\varphi \;\mid\; \varphi_1 \vee \varphi_2 \;\mid\; \langle c \rangle \varphi \;\mid\; \exists x(\varphi).$$

Given a model $\mathcal{M} = (W, I)$, an object $w \in W$ and an assignment function $s : \mathit{Var} \to W$, we define the interpretation of constants by:

$$[\![c]\!]_{w,s} = I(w)(c).$$

Additionally, the truth definition is given by:

$$\mathcal{M}, w, s \models \langle c \rangle \varphi \;\Leftrightarrow\; \mathcal{M}, v, s \models \varphi, \text{ where } v = I(w)(c).$$

The component I in (W, I) thus represents both the accessibility and the domain relation.

Consider for instance the model $\mathcal{M}$ in Figure 5, which consists of four objects that are arranged in a ring structure. Each object has two pointers *left* and *right* to denote the object on its left and on its right, respectively.

Figure 5: A ring structure

In this model, each object is the left neighbor of its right neighbor:

$$\mathcal{M} \models \exists x(x = \mathit{self} \wedge \langle \mathit{right} \rangle (\mathit{left} = x)).$$

Additionally, we have:

$$\mathcal{M} \models \exists x(x = \mathit{right} \wedge \langle \mathit{left} \rangle \langle \mathit{left} \rangle (\mathit{left} = x)),$$

which expresses that for each object, its right neighbor is the same object as the left neighbor of the left neighbor of its left neighbor.

We would like to end our discussion with the conclusion that our approach to the introduction of variable-binding mechanisms into modal logics provides a promising basis for various interesting extensions and applications.

Appendix: Proof of Theorem 17

First, we introduce the notion of an assignment-based bisimulation, which is almost similar to the notion of a history-based bisimulation. However, instead of injective sequences this type of bisimulation makes use of assignments.

Definition 20 Given the models $\mathcal{M}_1 = \langle W_1, R_1, D_1\rangle$ and $\mathcal{M}_2 = \langle W_2, R_2, D_2\rangle$, the worlds $w_1 \in W_1$ and $w_2 \in W_2$, the assignments $s_1 : Var \to W_1$ and $s_2 : Var \to W_2$, the non-empty relation $\sim$ is an *assignment-based bisimulation* if $(w_1, s_1) \sim (w_2, s_2)$ implies the following:

(term) $[\![t_1]\!]_{w_1,s_1} = [\![t_2]\!]_{w_1,s_1}$ iff $[\![t_1]\!]_{w_2,s_2} = [\![t_2]\!]_{w_2,s_2}$, for all terms t_1 and t_2.

(bisim′) if $u_1 \in R_1(w_1)$ then there exists $u_2 \in R_2(w_2)$ with $(u_1, s_1) \sim (u_2, s_2)$

(var′) if $u_1 \in D_1(w_1)$ then there is $u_2 \in D_2(w_2)$ with $(w_1, s_1[u_1/x]) \sim (w_2, s_2[u_2/x])$, for all $x \notin ran(s_1) \cap ran(s_2)$.

and conditions similar to **(bisim′)** and **(var′)** from $\mathcal{M}_2$ to $\mathcal{M}_1$.

For assignment-based bisimulations we have the following characterization result.

Lemma 21 Given the models $\mathcal{M}_1 = \langle W_1, R_1, D_1\rangle$ and $\mathcal{M}_2 = \langle W_2, R_2, D_2\rangle$, the worlds $w_1 \in W_1$ and $w_2 \in W_2$, the assignments $s_1 : Var \to W_1$ and $s_2 : Var \to W_2$, we have:

(*i*) if $(w_1, s_1) \sim (w_2, s_2)$ for some assignment-based bisimulation $\sim$ then for all $\varphi \in \mathcal{L}_1$ we have: $\mathcal{M}_1, w_1, s_1 \models \varphi \Leftrightarrow \mathcal{M}_2, w_2, s_2 \models \varphi$

(*ii*) if w_1 and w_2 are image finite and $\mathcal{M}_1, w_1, s_1 \models \varphi \Leftrightarrow \mathcal{M}_2, w_2, s_2 \models \varphi$ for all $\varphi \in \mathcal{L}_1$, then $(w_1, s_1) \sim (w_2, s_2)$ for some assignment-based bisimulation $\sim$.

Proof (*i*) This can be shown by induction on the complexity of φ. We consider the main case: φ is of the form $\exists x\psi$. Suppose $(w_1, s_1) \sim (w_2, s_2)$ and $\mathcal{M}, w_1, s_1 \models \exists x\psi$. Then there exists $u_1 \in D_1(w_1)$ such that $\mathcal{M}, w_1, s_1[u_1/x] \models \psi$ holds. From condition **(var′)** we derive that there exists $u_2 \in D_2(w_2)$ with $(w_1, s_1[u_1/x]) \sim (w_2, s_2[u_2/x])$. Applying the induction hypothesis, we obtain $\mathcal{M}_2, w_2, s_2[u_2/x] \models \psi$ for some $u_2 \in D_2(w_2)$, which yields the desired result $\mathcal{M}_2, w_2, s_2 \models \exists x\psi$. The converse implication can be shown similarly.

(*ii*) Consider the relation $\sim$ defined by: $(w_1, s_1) \sim (w_2, s_2)$ iff w_1 and w_2 are image finite and $\mathcal{M}_1, w_1, s_1 \models \varphi \;\Leftrightarrow\; \mathcal{M}_2, w_2, s_2 \models \varphi$ for all formulas $\varphi \in \mathcal{L}_1$. We claim that $\sim$ is an assignment-based bisimulation. Suppose this is not the case. Then there exist image-finite worlds w_1 and w_2 and assignments s_1 and s_2 with $\mathcal{M}_1, w_1, s_1 \models \varphi \;\Leftrightarrow\; \mathcal{M}_2, w_2, s_2 \models \varphi$ for all formulas $\varphi \in \mathcal{L}_1$, but at least one of the conditions of assignment-based bisimulations is not satisfied.

- First of all suppose **(term)** is not satisfied. Then without loss of generality, for some terms t_1 and t_2 we have $[\![t_1]\!]_{w_1,s_1} = [\![t_2]\!]_{w_1,s_1}$ but $[\![t_1]\!]_{w_2,s_2} \neq [\![t_2]\!]_{w_2,s_2}$. In other words, $\mathcal{M}_1, w_1, s_1 \models (t_1 = t_2)$ and $\mathcal{M}_2, w_2, s_2 \not\models (t_1 = t_2)$, which yields a contradiction.

- Next, we assume that condition **(bisim′)** is not satisfied. Consider the set $R_2(w_2) = \{u_1, \dots, u_k\}$ of worlds that are R_2-accessible from w_2, which is finite because of the image-finiteness of w_2. Suppose that we have a world $v \in R_1(w_1)$ such that for all $1 \leq i \leq k$ there exists a formula $\varphi_i \in \mathcal{L}_1$ with $\mathcal{M}_1, v, s_1 \models \varphi_i$ and $\mathcal{M}_2, u_i, s_2 \not\models \varphi_i$. Let φ be the conjunction $\bigwedge_{1\leq i\leq k} \varphi_i$. Then we have $\mathcal{M}_1, w_1, s_1 \models \Diamond\varphi$ while $\mathcal{M}_2, w_2, s_2 \not\models \Diamond\varphi$, yielding a contradiction. We conclude that such a world v cannot exist.

- Finally, we suppose that condition **(var′)** is not met. Consider the set $D_2(w_2) = \{u_1, \dots, u_k\}$, which is finite by the image finiteness condition. Suppose we have a world $v \in D_1(w_1)$ such that for all $1 \leq i \leq k$ there exists a formula $\varphi_i \in \mathcal{L}_1$ with $\mathcal{M}_1, w_1, s_1[v/x] \models \varphi_i$ and $\mathcal{M}_2, w_2, s_2[u_i/x] \not\models \varphi_i$. Subsequently, for the conjunction $\varphi = \bigwedge_{1\leq i\leq k} \varphi_i$ we obtain $\mathcal{M}_1, w_1, s_1 \models \exists x\varphi$ and $\mathcal{M}_2, w_2, s_2 \not\models \exists x\varphi$, yielding a contradiction.

Hence, we conclude that $\sim$ is an assignment-based bisimulation.

What remains to be done is the translation of these results to the case of history-based bisimulations. First, we consider the connection between sequences and assignments. Recall that we assume sequences to be *injective*, which means that they do not contain any repetitions. We define the relation $\approx$, which relates pairs of sequences with the pairs of assignments they represent.

Definition 22 For all sequences $\bar{w_1}$ and $\bar{w_2}$ and assignments s_1 and s_2, we define $(\bar{w_1}, \bar{w_2}) \approx (s_1, s_2)$ if and only if the following hold:

(*i*) $ran(s_1) = ran(\bar{w_1})$ and $ran(s_2) = ran(\bar{w_2})$

(*ii*) $s_1(x) = \bar{w_1}(i)$ iff $s_2(x) = \bar{w_2}(i)$, for all $x \in Var$ and $i \in I\!N$.

We identify the following properties of the relation $\approx$.

Proposition 23 For all sequences w_1 and w_2, assignments s_1 and s_2, if $(\bar{w_1}, \bar{w_2}) \approx (s_1, s_2)$ then:

(*i*) $s_1(x) = s_1(y) \;\Leftrightarrow\; s_2(x) = s_2(y)$, for all $x, y \in Var$

(*ii*) $(\bar{w_1} \bullet u_1, \bar{w_2} \bullet u_2) \approx (s_1[u_1/x], s_2[u_2/x])$, for all $x \notin dom(s_1) \cap dom(s_2)$.

The following result establishes how to construct history-based bisimulations from assignment-based bisimulations and vice versa.

Lemma 24

(*i*) If $\sim$ is an assignment-based bisimulation then the following relation $\sim_h$ is a history-based bisimulation:

$(w_1, \bar{v_1}) \sim_h (w_2, \bar{v_2})$ iff $(w_1, s_1) \sim (w_2, s_2)$ for some s_1, s_2 with $(\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$

(*ii*) If $\sim$ is a history-based bisimulation then the following relation $\sim_a$ is an assignment-based bisimulation:

$$(w_1, s_1) \sim_a (w_2, s_2) \text{ iff } (w_1, \bar{v_1}) \sim (w_2, \bar{v_2}) \text{ for some } \bar{v_1}, \bar{v_2} \text{ with } (\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$$

Proof (*i*) Suppose $(w_1, \bar{v_1}) \sim_h (w_2, \bar{v_2})$, where $\sim$ is an assignment-based bisimulation. By definition there exist assignments s_1, s_2 with $(\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$ and $(w_1, s_1) \sim (w_2, s_2)$. We have to prove that all conditions for history-based bisimulations are satisfied.

(self) If we take the constant *self* for t_1 and a variable x for t_2 in **(term)** we derive $w_1 = s_1(x) \Leftrightarrow w_2 = s_2(x)$, for all x. From the fact $(\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$ we derive $w_1 = \bar{v_1}(i) \Leftrightarrow w_2 = \bar{v_2}(i)$ for all $i \in \mathbb{N}$, which was to be shown.

(bisim) This condition follows immediately from **(bisim′)**.

(var) Consider a state $u_1 \in D_1(w_1)$. By condition **(var′)** there exists $u_2 \in D_2(w_2)$ with $(w_1, s_1[u_1/x]) \sim (w_2, s_2[u_2/x])$, for all $x \notin ran(s_1) \cap ran(s_2)$. Additionally, from $(\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$ we derive via Proposition 23(*ii*) that $(\bar{v_1} \bullet u_1, \bar{v_2} \bullet u_2) \approx (s_1[u_1/x], s_2[u_2/x])$ holds. Consequently, we have $(w_1, \bar{v_1} \bullet u_1) \sim_h (w_2, \bar{v_2} \bullet u_2)$, which was to be shown.

(*ii*) Suppose $(w_1, s_1) \sim_a (w_2, s_2)$, where $\sim$ is a history-based bisimulation. By definition there exist $\bar{v_1}$ and $\bar{v_2}$ with $(\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$ and $(w_1, \bar{v_1}) \sim (w_2, \bar{v_2})$. We have to prove that all conditions for assignment-based bisimulations are satisfied.

(term) First, condition **(self)** gives $w_1 = \bar{v_1}(i) \Leftrightarrow w_2 = \bar{v_2}(i)$ for all i. From the fact $(\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$ we derive $w_1 = s_1(x) \Leftrightarrow w_2 = s_2(x)$, for all $x \in Var$. Secondly, Proposition 23(*i*) yields $s_1(x) = s_1(y) \Leftrightarrow s_2(x) = s_2(y)$, for all $x, y \in Var$. Together these two facts yield $[\![t_1]\!]_{w_1,s_1} = [\![t_2]\!]_{w_1,s_1}$ iff $[\![t_1]\!]_{w_2,s_2} = [\![t_2]\!]_{w_2,s_2}$, for all terms t_1 and t_2.

(bisim′) This condition follows immediately from **(bisim)**.

(var′) Consider a state $u_1 \in D_1(w_1)$. By condition **(var)** there exists $u_2 \in D_2(w_2)$ with $(w_1, \bar{v_1} \bullet u_1) \sim (w_2, \bar{v_2} \bullet u_2)$. Additionally, from $(\bar{v_1}, \bar{v_2}) \approx (s_1, s_2)$ we derive via Proposition 23(*ii*) that $(\bar{v_1} \bullet u_1, \bar{v_2} \bullet u_2) \approx (s_1[u_1/x], s_2[u_2/x])$ holds for all $x \notin ran(s_1) \cap ran(s_2)$. Consequently, we have $(w_1, s_1[u_1/x]) \sim (w_2, s_2[u_2/x])$ for all $x \notin ran(s_1) \cap ran(s_2)$, which was to be shown.

Finally, we are in position to put all pieces together.

Proof of Theorem 17
(*i*) Suppose $w_1 \sim w_2$ for some history-based bisimulation $\sim$, which means $(w_1, \epsilon) \sim (w_2, \epsilon)$. Then according to Lemma 24 we have $(w_1, s) \sim_a (w_2, s)$ for the assignment-based bisimulation $\sim_a$, where s is defined by $s(x) = \bot$ for all $x \in Var$. Then Lemma 21 yields that for all formulas $\varphi \in \mathcal{L}_1$ we have $\mathcal{M}_1, w_1, s \models \varphi \Leftrightarrow \mathcal{M}_1, w_2, s \models \varphi$. Consequently, for all sentences $\varphi \in \mathcal{L}_1$ we derive $\mathcal{M}_1, w_1 \models \varphi \Leftrightarrow \mathcal{M}_1, w_2 \models \varphi$.

(*ii*) Suppose w_1 and w_2 are image finite and $\mathcal{M}_1, w_1 \models \varphi \Leftrightarrow \mathcal{M}_2, w_2 \models \varphi$, for all sentences $\varphi \in \mathcal{L}_1$. Since we restrict to sentences we also have $\mathcal{M}_1, w_1, s \models \varphi \Leftrightarrow \mathcal{M}_2, w_2, s \models \varphi$ for all $\varphi \in \mathcal{L}_1$, where s is defined by $s(x) = \perp$ for all $x \in \mathit{Var}$. By Lemma 21 we have $(w_1, s) \sim (w_2, s)$ for some history-based bisimulation $\sim$. Lemma 24 then establishes $(w_1, \epsilon) \sim_h (w_2, \epsilon)$, for the history-based bisimulation $\sim_h$, and thus we conclude $w_1 \sim_h w_2$.

Acknowledgments

The authors wish to thank the anonymous referees for their suggestions and comments.

References

[1] H. Andréka, J. van Benthem, and I. Németi. Modal logics and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27(3):217–274, 1999.

[2] C. Areces, P. Blackburn, and M. Marx. A road-map on the complexity of hybrid logics. In J. Flum and M. Rodríguez-Artalejo, editors, *Computer Science Logic, Proceedings of CSL'99*, volume 1683 of *Lecture Notes in Computer Science*, pages 307–321. Springer-Verlag, Heidelberg, 1999.

[3] J.F.A.K van Benthem. *Modal Logic and Classical Logic.* Bibliopolis, Naples, 1983.

[4] P. Blackburn and J. Seligman. Hybrid languages. *Journal of Logic, Language and Information*, 4:251–272, 1995.

[5] R. Bull. An approach to tense logic. *Theoria*, 36:282–306, 1970.

[6] R.M. van Eijk, F.S. de Boer, W. van der Hoek, and J.-J.Ch. Meyer. A modal logic for network topologies. In M. Ojeda-Aciego, I.P. de Guzman, G. Brewka, and L.M. Pereira, editors, *Proceedings of the 7th European Workshop on Logics in Artificial Inteligence (JELIA 2000)*, volume 1919 of *Lecture Notes in Artificial Intelligence*, pages 269–283. Springer-Verlag, Heidelberg, 2000.

[7] M. Fitting and R.L. Mendelsohn. *First-Order Modal Logic.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[8] U. Goltz, R. Kuiper, and W. Penczek. Propositional temporal logics and equivalences. In *Proceedings of Concur'92*, volume 630 of *Lecture Notes in Computer Science*, pages 222–236, Berlin, 1992. Springer-Verlag.

[9] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of Association of Computer Machinery*, 32:137–162, 1985.

[10] W. van der Hoek. On the semantics of graded modalities. *Journal of Applied Non Classical Logics*, 2(1):81–123, 1992.

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

Resolution in Modal, Description and Hybrid Logic

Carlos Areces[1] Hans de Nivelle[2]
Maarten de Rijke[1]

[1]ILLC, University of Amsterdam
Nieuwe Achtergracht 166, 1018 WV Amsterdam, The Netherlands
Email: {carlos,mdr}@science.uva.nl
[2]Max Planck Institut für Informatik
Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
Email: nivelle@mpi-sb.mpg.de

Abstract

We provide a resolution-based proof procedure for modal, description and hybrid logic that improves on previous proposals in important ways. It avoids translations into large undecidable logics, and works directly on modal, description or hybrid logic formulas instead. In addition, by using the hybrid machinery it avoids the complexities of earlier propositional resolution-based methods for modal logic. It combines ideas from the method of prefixes used in tableaux, and resolution ideas in such a way that some of the heuristics and optimizations devised in either field are applicable.

Keywords: direct resolution, modal logic, description logic, hybrid logic.

1 Introduction

Resolution, originally introduced for first-order logic (FO) in [36], is the most widely used reasoning method for first-order logic today: most of the available automatic theorem provers for FO are resolution based. The propositional core of the method is well-known: to check whether a propositional formula ϕ is not satisfiable, start by turning it into *clausal form.* To this end, we write ϕ in conjunctive normal form

$$\phi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)},$$

for $\psi_{(l,m)}$ a literal, and let the clause set associated with ϕ be

$$\mathit{ClSet}(\phi) = \{\{\psi_{(l,m)} \mid m \in M\} \mid l \in L\}.$$

Next, we define $\mathit{ClSet}^*(\phi)$ as the smallest set containing $\mathit{ClSet}(\phi)$ and closed under a unique, very easy to grasp rule:

$$\text{(RES)} \quad \frac{\mathit{Cl}_1 \cup \{N\} \in \mathit{ClSet}^*(\phi) \quad \mathit{Cl}_2 \cup \{\neg N\} \in \mathit{ClSet}^*(\phi)}{\mathit{Cl}_1 \cup \mathit{Cl}_2 \in \mathit{ClSet}^*(\phi)}$$

If $\{\} \in ClSet^*(\phi)$, then ϕ is not satisfiable. The intuition behind (RES) is as follows: given that either N or $\neg N$ is always the case in any model, they can be "cut away" if the sets of clauses are conjoined. The aim of the whole method is to "cut away everything" and arrive at the empty set.

The elegance of the resolution method for propositional logic relies mostly on its bare simplicity. The method can also be straightforwardly implemented, it seems tailored for a dumb machine able to crunch symbols quickly. The only computational cost is a search for complementary atoms in the set of clauses.

Of course, the picture for the first-order case is different (to start with, first-order resolution has to address an undecidable problem!). And actual implementations of first-order resolution systems are not "dumb" at all. In particular, during first-order resolution we have to cope with the rich structure of terms, and the unification algorithm (introduced by Robinson in [36], see [33] for a linear time version) plays a fundamental role in handling this complexity, and in using it to guide the search. The field of resolution based first-order theorem proving has developed into a community of its own, with an impressive collection of methods and optimizations [9, 34].

In contrast to the popularity of resolution-based methods in first-order logic, modern *modal* theorem provers are generally based on tableau methods [15]. Nowadays, resolution and modal languages seem to be related only when *indirect* methods are used. In translation-based resolution calculi for modal logics, one translates modal languages into a large background language (typically first-order logic), and devises strategies that guarantee termination for the fragment corresponding to the original modal language [22, 29, 17, 5]. First-order resolution provers like BLIKSEM [12] or SPASS [38] handle modal formulas in this way, in some cases using extremely optimized translations like those investigated in [32, 37]. This approach has both advantages and disadvantages with respect to the tableau approach. On the one hand we can translate many systems into the same background language and hence explore different, and also combined, systems without the need to modify the prover. But empirical tests show that the price to pay is high [28, 5]. The undecidability of the full background language shows up in degraded performance on the modal fragments, and first-order provers can hardly emulate their tableau based competitors.

Given the simplicity of propositional resolution, it is natural to wonder why *direct* resolution methods for modal languages don't figure in the picture. Designing resolution methods that can directly (without translation into large background languages) be applied to modal logics, received some attention in the late 1980s and early 1990s [20, 30, 16]. Also, the first (non-clausal) resolution methods for temporal languages go back to that period with the work of Abadi and Manna [1]. Recently, new results on clausal temporal resolution have been presented (see [18]). But even though we might sometimes think of modal languages as a "simple extension of propositional logic," direct resolution for modal languages has proved a difficult task. Intuitively, in basic modal languages the resolution rule has to operate *inside* boxes and diamonds to achieve completeness. This leads to more complex systems, less elegant results, and poorer performance, ruining the one-dumb-rule spirit of resolution.

In this paper we will show how ideas from hybrid logics can be put to work with benefit even when the subject is purely modal. In particular, aided by the notions of nominals and labeling, we will show how to define simple direct resolution methods for modal languages. This "case study" is an example of how the additional flexibility provided by the ability to name states can be used to improve reasoning methods. In addition, we can build over the basic resolution system and obtain extensions for hybrid and also description languages.

The main characteristics of the resolution method we will introduce can be summarized as follows:

- by using labeled formulas it avoids the complexity of earlier direct resolution-based methods for modal logic;
- it does not involve skolemization beyond the use of constants;
- it does not involve translation into large undecidable languages, working directly on modal, hybrid or description logic formulas instead;
- it is flexible and conservative in more than one sense: it incorporates the method of prefixes used in tableaux [23] into resolution in such a way that different heuristics and optimizations devised in either field are applicable.

The structure of the paper is as follows. In Section 2 we discuss, in some detail, the problems with direct modal resolution. We do this by discussing the system presented by Enjalbert and Fariñas del Cerro in [20]. In Section 3 we introduce *labeled resolution* for the basic multi-modal logic $\mathbf{K}_m$. Informally, the calculus uses the hybrid @ operator to "push formulas out of modalities" and in this way, feed them into a simple (RES) rule. Whereas in Section 3, we use the hybrid machinery only at the meta-logical level, the next step naturally leads us to internalized systems. We start in Section 4, by providing a resolution based method for deciding knowledge base inconsistency (with simple, acyclic T-Boxes and non-empty A-Boxes) for the description logic $\mathcal{ALCR}$; as discussed in [2], $\mathcal{ALCR}$ can be viewed as a restricted hybrid language. In Section 5 we finally move into the complete basic hybrid language, fully internalizing our use of @ and explaining how to handle nominals. We need to incorporate a form of equality reasoning into the resolution calculus, and discuss paramodulation and other techniques. In the last part of this section we show how to treat very expressive hybrid languages, by considering the $\downarrow$ hybrid binder. In Section 6 we conclude, with comments on related work and some directions for further research.

2 Direct Resolution for Modal Languages

To understand how we can use hybrid logic ideas to improve direct modal resolution, we introduce the system presented by Enjalbert and Fariñas del Cerro in [20]. Enjalbert and Fariñas del Cerro use some non-standard definitions which we introduce below and to which we adhere only in the present section; we will revert to more standard notation in the rest of the paper.

A modal formula is in *disjunctive normal form* if it is a (possibly empty) disjunction of the form

$$\phi = \bigvee L_i \vee \bigvee \Box D_j \vee \bigvee \Diamond A_k,$$

where each L_i is a literal, each D_j is in disjunctive normal form, and each A_k is in conjunctive normal form. A modal formula is in *conjunctive normal form* if it is a

3

conjunction $\phi = \bigwedge C_i$, where each C_i is in disjunctive normal form. A formula in disjunctive normal form is called a *clause*. The empty clause is denoted as $\bot$. The conjunction $C_1 \wedge \cdots \wedge C_n$ is identified with the set $(C_1, \dots, C_n)$. For any modal formula an equivalent clause can be obtained, so that attention can be restricted to clauses.

The following examples of applications of the resolution rule "in modal contexts" are discussed in [20] to show the intricacies of modal resolution:

$$(a)\quad \frac{\Box(p \vee q) \qquad \Box\neg p}{\Box q} \qquad\qquad (b)\quad \frac{\Box(p \vee q) \qquad \Diamond\neg p}{\Diamond(\neg p, q)}.$$

Both inferences are sound, and can be viewed as generalizations of the (RES) rule. While (a) closely follows the (RES) pattern (we resolve on p inside $\Box$ and cut it out to obtain $\Box q$), (b) is more complex: we again resolve on p but simply eliminating p from $\Box(p \vee q)$ to obtain $\Box q$ is unsound. Instead, we can soundly infer $\Diamond q$ which would somehow follow the "cutting" pattern of resolution but this is too weak; the proper inference being both $\neg p$ and q possible at the same state of the model.

Moreover, an attempt to apply a similar rule to the clauses $\Diamond(p \vee q)$ and $\Diamond\neg p$ to derive $\Diamond(\neg p, q)$ does not preserve soundness. Also, inferences with only one premise seem to be needed, as for example in

$$(c)\quad \frac{\Diamond(\neg p, p \vee q)}{\Diamond(\neg p, p \vee q, q)},$$

where we resolve on p inside the *same* clause to infer $\Diamond(\neg p, q)$. Enjalbert and Fariñas del Cerro explain that, actually, the logically equivalent but more explicit formula $\Diamond(\neg p, p \vee q, q)$ needs to be retained for completeness of the resolution method.

A resolution system based on these intuitions has been introduced and proved complete for **K** in [20]. Specifically, define, by induction, two relations on clauses $\Sigma(\alpha, \beta) \to \gamma$ and $\Gamma(\alpha) \to \gamma$, as indicated in Figure 1, where α, β, κ, δ_1, δ_2 are clauses, and Ψ, Φ are sets of clauses. The relations Σ and Γ will be used to define the notion

Axioms	
(A1) $\Sigma(p, \neg p) \to \bot$ (A2) $\Sigma(\bot, \alpha) \to \bot$	
Σ-Rules	**Γ-Rules**
$(\vee)\ \dfrac{\Sigma(\alpha,\beta) \to \kappa}{\Sigma(\alpha \vee \delta_1, \beta \vee \delta_2) \to \kappa \vee \delta_1 \vee \delta_2}$	$(\Diamond 1)\ \dfrac{\Sigma(\alpha,\beta) \to \kappa}{\Gamma(\Diamond(\alpha,\beta,\Phi)) \to \Diamond(\alpha,\beta,\kappa,\Phi)}$
$(\Box\Diamond)\ \dfrac{\Sigma(\alpha,\beta) \to \kappa}{\Sigma(\Box\alpha, \Diamond(\beta,\Psi)) \to \Diamond(\beta,\kappa,\Psi)}$	$(\Diamond 2)\ \dfrac{\Gamma(\alpha) \to \beta}{\Gamma(\Diamond(\alpha,\Phi)) \to \Diamond(\beta,\alpha,\Phi)}$
$(\Box\Box)\ \dfrac{\Sigma(\alpha,\beta) \to \kappa}{\Sigma(\Box\alpha, \Box\beta) \to \Box\kappa}$	$(\vee)\ \dfrac{\Gamma(\alpha) \to \beta}{\Gamma(\alpha \vee \kappa) \to \beta \vee \kappa}$
	$(\Box)\ \dfrac{\Gamma(\alpha) \to \beta}{\Gamma(\Box\alpha) \to \Box\beta}$

Figure 1: Enjalbert and Fariñas del Cerro resolution rules.

of resolvent, i.e., of a clause obtained via resolution from a previous set of clauses. The definition is rather involved, starting from the axioms stating the cut on opposing literals and the propagation of inconsistencies ((A1) and (A2)), to the inductive steps which specify how disjunctions should be handled (the pair of ($\vee$) rules) and how one should deal with modal contexts (($\Box\Diamond$), ($\Box\Box$), ($\Diamond 1$), ($\Diamond 2$), ($\Box$)).

The full formal definition runs as follows. Start by defining the simplification relation $A \approx B$ (perhaps better understood as a rewriting system $\rightsquigarrow$) as the least congruence containing

$$\begin{array}{rcl|rcl} \Diamond\bot & \approx & \bot & \bot \vee D & \approx & D \\ (\bot, A) & \approx & \bot & A \vee A \vee D & \approx & A \vee D. \end{array}$$

For any formula F there is a unique F' such that $F \approx F'$ and F' cannot be simplified further. This formula F' is called the normal form of F. C is a *resolvent* of A and B (respectively A) iff there is some C' such that $\Sigma(A, B) \to C'$ (respectively, $\Gamma(A) \to C'$) and C is the normal form of C'. We write $\Sigma(A, B) \Rightarrow C$ (respectively, $\Gamma(A) \Rightarrow C$) if C is a resolvent of A and B (respectively, of A).

Given a set of clauses S, let $\mathit{ClSet}^*(S)$ be the smallest set containing S that is closed under resolvents of elements in $\mathit{ClSet}^*(S)$. D is said to be a *resolution consequence* of a set of clauses S (notation $S \vdash D$) iff $D \in \mathit{ClSet}^*(S)$.

Theorem 1 ([20]) For $S \cup \{D\}$ a finite set of clauses, $S \vdash D$ iff $\models_{\mathbf{K}} \bigwedge S \to D$.

So much for the "one dumb rule spirit" of resolution. Let us go through an example to better understand how this resolution method works. As is standard, we use the resolution system to check for unsatisfiability. If starting from a clause ϕ we are able to derive the empty clause $\bot$, then ϕ is unsatisfiable.

Example 2 Consider the formula $\Diamond(p \wedge (\neg p \vee \Box r \vee q)) \wedge \Box\neg q \wedge \Box\Diamond\neg r$. In the resolution proof below we underline the literals on which resolution takes place, and simplify some steps for succinctness.

1. $(\Diamond(\underline{p}, \underline{\neg p} \vee \Box r \vee q), \Box\neg q, \Box\Diamond\neg r)$ by (A1), ($\vee$) and ($\Diamond 1$)
2. $(\Diamond(p, \neg p \vee \Box r \vee q, \Box r \vee \underline{q}), \Box\underline{\neg q}, \Box\Diamond\neg r)$ by (A1), ($\vee$) and ($\Box\Diamond$)
3. $(\Diamond(p, \neg p \vee \Box r \vee q, \Box r \vee q, \Box\underline{r}), \Box\neg q, \Box\Diamond\underline{\neg r})$ by (A1) and ($\Box\Diamond$) twice
4. $(\Diamond(p, \neg p \vee \Box r \vee q, \Box r \vee q, \Box r, \Diamond(\neg r, \bot)), \Box\neg q, \Box\Diamond\neg r)$ by (A2) and ($\Diamond 1$)
5. $\bot$

Even following this simple proof is complex. For example, line 1 should be understood as follows. Given that $(p, \neg p) \Rightarrow \bot$ by (A1), we can infer by ($\vee$) that $(p, \neg p \vee \Box r \vee q) \Rightarrow (p, \neg p \vee \Box r \vee q, \Box r \vee q)$ (this already involves some simplifications). An application of ($\Diamond 1$) allows us to perform this inference under $\Diamond$.

As we have just seen, the direct resolution method for modal logics presented in [20] (and, similarly, those in [21, 30, 16]) performs resolution "inside" modalities, leading to a proliferation of deductive rules. In the next sections we develop a direct resolution method for modal, description and hybrid logic that retains as much of the lean one-rule character of traditional resolution methods as possible. The key idea, from a basic modal logic perspective, is to use labels to decorate formulas with additional information. Labels allow us to make information explicit and resolution can then always be performed at the "top level." From a hybrid logic perspective, we are just taking advantage of the new expressive power that nominals and @ provide.

3 Labeled Modal Resolution

We now introduce a direct resolution proof procedure for the basic multi-modal logic $\mathbf{K}_m$. We assume a fixed modal similarity type $\mathcal{S} = \langle \mathsf{REL}, \mathsf{PROP} \rangle$ of accessibility relation and propositional symbols, together with a basic hybrid logic similarity type $\mathcal{S}' = \langle \mathsf{REL}, \mathsf{PROP}, \mathsf{NOM} \rangle$, where NOM is a countably infinite set of nominals (we use ATOM to denote $\mathsf{PROP} \cup \mathsf{NOM}$).

Definition 3 (Normal Form) We define the following rewriting procedure *nf* on modal formulas:

$$\begin{array}{rcl} \neg\neg\phi & \overset{nf}{\rightsquigarrow} & \phi, \\ \langle R \rangle \phi & \overset{nf}{\rightsquigarrow} & \neg([R]\neg\phi), \\ (\phi_1 \vee \phi_2) & \overset{nf}{\rightsquigarrow} & \neg(\neg\phi_1 \wedge \neg\phi_2). \end{array}$$

For any formula ϕ, the rewriting of subformulas of ϕ by means of $\overset{nf}{\rightsquigarrow}$ converges to a unique *normal form* $\mathit{nf}(\phi)$ which is logically equivalent to ϕ. If we take $\vee$ and $\langle R \rangle$ as defined operators, then $\mathit{nf}(\phi)$ is slightly more than an expansion of definitions (see [27]).

Definition 4 (Clauses) A *clause* is a finite set Cl such that each element of Cl is a formula of the form $t:\phi$ or $(t_1, t_2):R$ for $t, t_1, t_2 \in \mathsf{NOM}$, $R \in \mathsf{REL}$ and ϕ a basic multi-modal formula. Let ϕ be a basic multi-modal formula. The set S_ϕ of clauses corresponding to ϕ is simply $\{\{a : \mathit{nf}(\phi)\}\}$, for a an arbitrary label in NOM.

Formulas in clauses can be seen as *labeled formulas* [23, 25]; $t_1:\phi$ specifies that the formula ϕ holds at the label t_1, and $(t_1, t_2) : R$ requires the labels t_1 and t_2 to be related by the accessibility relation R. Equivalently, a set of clauses can be seen as a set of hybrid formulas, with $t : \phi$ standing for $@_t\phi$ and $(t_1, t_2) : R$ standing for $@_{t_1}\langle R \rangle t_2$, and we can use hybrid models to define satisfiability for clauses. We recall the definition of hybrid model and satisfaction (see [2] for details).

Definition 5 (Semantics) A (hybrid) *model* $\mathcal{M}$ is a triple $\mathcal{M} = \langle M, \{R_i\}, V \rangle$ such that M is a non-empty set, $\{R_i\}$ is a set of binary relations on M, and $V : \mathsf{PROP} \cup \mathsf{NOM} \rightarrow Pow(M)$ is such that for all nominals $i \in \mathsf{NOM}$, $V(i)$ is a singleton subset of M. Let $\mathcal{M} = \langle M, \{R_i\}, V \rangle$ be a model and $m \in M$. The relevant conditions for the *satisfiability relation* are defined as follows:

$$\begin{array}{rcl} \mathcal{M}, m \Vdash a & \text{iff} & m \in V(a),\ a \in \mathsf{ATOM} \\ \mathcal{M}, m \Vdash [R]\phi & \text{iff} & \forall m'.(\text{if } R(m, m') \text{ then } \mathcal{M}, m' \Vdash \phi) \\ \mathcal{M}, m \Vdash @_i\phi & \text{iff} & \mathcal{M}, m' \Vdash \phi, \text{ where } V(i) = \{m'\},\ i \in \mathsf{NOM}. \end{array}$$

If $\mathcal{M}$ is understood from the context, we simply write $m \Vdash \phi$ for $\mathcal{M}, m \Vdash \phi$. We write $\mathcal{M} \Vdash \phi$ iff for all $m \in M$, $\mathcal{M}, m \Vdash \phi$.

Definition 6 (Satisfiability of Clauses) Let Cl be a clause, and let $\mathcal{M}$ be a hybrid model. We write $\mathcal{M} \models Cl$ if $\mathcal{M} \models \bigvee Cl$. A set of clauses S is *satisfiable* if there is a model $\mathcal{M}$ such that for all $Cl \in S$, $\mathcal{M} \models Cl$.

Let ϕ be a basic multi-modal formula and S_ϕ its corresponding set of clauses. Proving that ϕ is satisfiable iff S_ϕ is satisfiable is straightforward. For the left to right implication, given $\mathcal{M} = \langle W, \{R_i\}, V\rangle$ and $m \in M$ such that $\mathcal{M}, m \Vdash \phi$, just extend V so that $V(a) = \{m\}$ and give any interpretation to others elements in NOM. For the other direction, drop the interpretation of elements in NOM.

We have now set up the machinery to provide the appropriate set of resolution rules. As a guide, it is useful to recall that modal formulas can be seen as first-order formulas by means of the standard translation ST.

Definition 7 (Standard Translation into FO) The mutually recursive functions ST_x and ST_y map basic modal formulas into FO:

$$\begin{array}{r@{\;}l|r@{\;}l}
ST_x(p_j) &= P_j(x),\ p_j \in \mathsf{PROP} & ST_y(p_j) &= P_j(y),\ p_j \in \mathsf{PROP}\\
ST_x(\neg\phi) &= \neg ST_x(\phi) & ST_y(\neg\phi) &= \neg ST_y(\phi)\\
ST_x(\phi\wedge\psi) &= ST_x(\phi)\wedge ST_x(\psi) & ST_y(\phi\wedge\psi) &= ST_y(\phi)\wedge ST_y(\psi)\\
ST_x([R]\phi) &= \forall y.(R(x,y)\rightarrow ST_y(\phi)) & ST_y([R]\phi) &= \forall x.(R(y,x)\rightarrow ST_x(\phi)).
\end{array}$$

Figure 2 provides a set of rules transforming sets of clauses into sets of clauses.

$$(\wedge)\quad \dfrac{Cl\cup\{t\!:\!\phi_1\wedge\phi_2\}}{\begin{array}{c}Cl\cup\{t\!:\!\phi_1\}\\ Cl\cup\{t\!:\!\phi_2\}\end{array}} \qquad (\neg\wedge)\quad \dfrac{Cl\cup\{t\!:\!\neg(\phi_1\wedge\phi_2)\}}{Cl\cup\{t\!:\!nf(\neg\phi_1), t\!:\!nf(\neg\phi_2)\}}$$

$$(\mathrm{RES})\quad \dfrac{Cl_1\cup\{t\!:\!\phi\}\quad Cl_2\cup\{t\!:\!\neg\phi\}}{Cl_1\cup Cl_2}$$

$$([R])\quad \dfrac{Cl_1\cup\{t_1\!:\![R]\phi\}\quad Cl_2\cup\{(t_1,t_2)\!:\!R\}}{Cl_1\cup Cl_2\cup\{t_2\!:\!\phi\}}$$

$$(\neg[R])\quad \dfrac{Cl\cup\{t\!:\!\neg[R]\phi\}}{\begin{array}{c}Cl\cup\{(t,n)\!:\!R\}\\ Cl\cup\{n\!:\!nf(\neg\phi)\}\end{array}},\ \text{where } n \text{ is new.}$$

Figure 2: Labeled resolution rules.

If you read the rules with ST in the back of your mind, the meaning of ($[R]$) and ($\neg[R]$) will be immediately clear. ($[R]$) is needed to account for the “hidden” negation arising from the implication in the translation of $[R]$, and in that sense it is indeed a standard resolution rule which cuts away the two complementary binary literals $\neg R(t_1, x)$ and $R(t_1, t_2)$ and unifies x to t_2. On the other hand, ($\neg[R]$) can be seen as a mild kind of skolemization which only involves the introduction of constants. From this perspective, we can view the rules ($\wedge$), ($\neg\wedge$) and ($\neg[R]$) as preparing the input formula and feeding it into the resolution rules (RES) and ($[R]$). In other words, the system interleaves the reduction towards a standard clausal form with the resolution steps as in [24]. An immediate advantage of this method is that resolution can be performed not only on literals, but also on complex formulas.

Before moving on, let’s redo Example 2 in the new system. As before, we underline the part of the formula where a rule applies.

Example 8 Consider again the formula $\phi = \Diamond(p \wedge (\neg p \vee \Box r \vee q)) \wedge \Box\neg q \wedge \Box\Diamond\neg r$, which in the new notation is written as $\langle R\rangle(p \wedge (\neg p \vee [R]r \vee q)) \wedge [R]\neg q \wedge [R]\langle R\rangle\neg r$. S_ϕ is the singleton $\{\{i : \neg[R]\neg(p \wedge \neg(p \wedge \neg[R]r \wedge \neg q)) \wedge [R]\neg q \wedge [R]\neg[R]\neg r\}\}$. In each line we only show the newly generated clauses and those which will still be required in the successive steps.

1. $\{i : \neg[R]\neg(p \wedge \neg(p \wedge \neg[R]r \wedge \neg q))\underline{\triangle}[R]\neg q\underline{\triangle}[R]\neg[R]\neg r\}$, by $((\wedge)$ twice)
2. $\{i : \underline{\neg[R]}\neg(p \wedge \neg(p \wedge \neg[R]r \wedge \neg q))\}, \{i : [R]\neg q\},\ \{i : [R]\neg[R]r\}$, by $(\neg[R])$
3. $\{R(i,j)\}, \{j : (p\underline{\triangle}\neg(p \wedge \neg[R]r \wedge \neg q))\}, \{i : [R]\neg q\},\ \{i : [R]\neg[R]r\}$, by $(\wedge)$
4. $\{R(i,j)\}, \{j : p\}, \{j : \underline{\neg}(p\underline{\triangle}\neg[R]r\underline{\triangle}\neg q)\}, \{i : [R]\neg q\},\ \{i : [R]\neg[R]r\}$, by $(\neg\wedge)$
5. $\{R(i,j)\}, \{j : \underline{p}\}, \{j : \underline{\neg p}, j : [R]r, j : q\}, \{i : [R]\neg q\},\ \{i : [R]\neg[R]r\}$, by (RES)
6. $\{R(i,j)\}, \{j : [R]r, j : q\}, \{i : \underline{[R]}\neg q\},\ \{i : [R]\neg[R]r\}$, by $([R])$
7. $\{j : [R]r, j : \underline{q}\}, \{j : \underline{\neg q}\},\ \{j : \neg[R]r\}$, by (RES)
8. $\{j : \underline{[R]r}\},\ \{j : \underline{\neg[R]r}\}$, by (RES)
9. $\{\}$.

Definition 9 (Deduction) A *deduction* of a clause Cl from a set of clauses S is a finite sequence $S_1, \ldots, S_n$ of sets of clauses such that $S = S_1$, $Cl \in S_n$ and each S_i (for $i > 1$) is obtained from S_{i-1} by adding the consequent clauses of the application of one of the resolution rules in Figure 2 to clauses in S_{i-1}. Cl is a *consequence* of S if there is a deduction of Cl from S. A deduction of $\{\}$ from S is a *refutation* of S, and we say that S is *refutable*.

The set $\mathit{ClSet}^*(S)$, defined as the smallest set containing S and all its consequences, need not be finite because the rule $(\neg[R])$ can introduce infinitely many clauses which only differ on the label. By restricting $(\neg[R])$ to be "fired only once" as we will describe in the next section, we can ensure finiteness of $\mathit{ClSet}^*(S)$, and hence termination of the search for consequences.

It is straightforward to prove that the resolution rules in Figure 2 preserve satisfiability. That is, given a rule, if the premises are satisfiable, then so are the conclusions. In Section 4, we will extend the system to deal with knowledge bases in the description language $\mathcal{ALCR}$, and prove there, in detail, soundness, completeness and termination.

3.1 Modal Extensions

From a traditional modal logic point of view we often want to consider systems above $\mathbf{K}_m$. Here we choose systems **T**, **D**, and **4** as examples. Each system is axiomatically defined as an extension of the basic system **K** by the addition of an axiom scheme which characterizes certain property of the accessibility relation.

Name	Axiom Scheme	Accessibility Relation
T	$p \rightarrow \langle R\rangle p$	reflexivity: $\forall x.R(x,x)$
D	$[R]p \rightarrow \langle R\rangle p$	seriality: $\forall x\exists y.R(x,y)$
4	$\langle R\rangle\langle R\rangle p \rightarrow \langle R\rangle p$	transitivity: $\forall xyz.(R(x,y) \wedge R(y,z) \rightarrow R(x,z))$

Corresponding to each of the axioms we add a new resolution rule:

8

$$(\mathbf{T}) \quad \frac{Cl \cup \{t\!:\![R]\phi\}}{Cl \cup \{t\!:\!\phi\}}$$

$$(\mathbf{D}) \quad \frac{Cl \cup \{t\!:\![R]\phi\}}{Cl \cup \{t\!:\!\neg[R]nf(\neg\phi)\}}$$

$$(\mathbf{4}) \quad \frac{Cl_1 \cup \{t_1\!:\![R]\phi\} \quad Cl_2 \cup \{(t_1,t_2)\!:\!R\}}{Cl_1 \cup Cl_2 \cup \{t_2 : [R]\phi\}}.$$

Soundness for these systems is immediate. For completeness and termination we should modify the constructions in Section 4 (in particular (**4**) needs a mechanism of cycle detection); this can be done using methods from [20].

4 Description Logic

We will now discuss resolution based decision methods for description logics (DL's), a family of specialized languages for the representation and structuring of knowledge, related to the KL-ONE system of Brachman and Schmolze [13]. The connections between DL's and hybrid logics are strong (see [2, 3]), as we will clearly see in what follows. We spell out the details of a labeled resolution system to decide inconsistency of knowledge bases with so-called simple, acyclic T-Boxes and non-empty A-Boxes in the description logic $\mathcal{ALCR}$.

We assume fixed a description logic signature $\mathcal{S} = \langle \mathsf{CON}, \mathsf{ROL}, \mathsf{IND} \rangle$ together with an additional countable set of labels LAB. CON is the set of atomic concepts, ROL the set of atomic roles and IND the set of individuals.

Definition 10 (Interpretation) An *interpretation* $\mathcal{I}$ for $\mathcal{S}$ is a tuple $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where $\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is a function assigning an element $a_i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ to each individual a_i; a subset $C_i^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ to each atomic concept C_i; and a relation $R_i^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to each atomic role R_i.

In other words, a description logic interpretation is just a model for a particular kind of first-order signature, where only unary and binary predicate symbols are allowed and the set of function symbols is empty.

The atomic symbols in a description logic signature can be combined by means of *concept* and *role constructors*, to form complex concept and role expressions. Each description logic is characterized by the set of concept and roles constructors it allows. Figure 3 defines the roles and concepts constructors for the description logic $\mathcal{ALCR}$, together with their semantics.

In description logics we are interested in performing inferences given certain background knowledge.

Definition 11 (Knowledge Bases and Inference) A *knowledge base* is a pair $\Sigma = \langle T, A \rangle$ such that T is the T(erminological)-Box, a finite (possibly empty) set of *terminological axioms* of the form $C_1 \sqsubseteq C_2$ or $R_1 \sqsubseteq R_2$, and A is the A(ssertional)-Box, a finite (possibly empty) set of *assertions* of the form $a\!:\!C$ or $(a,b)\!:\!R$, where C, C_1, C_2 are complex concepts, R, R_1, R_2 are complex roles, and a, b are individuals.

For $\mathcal{I}$ an interpretation and ϕ a terminological axiom or an assertion, we define the relation $\mathcal{I} \models \phi$ as follows

Constructor	Syntax	Semantics
concept name	C	$C^{\mathcal{I}}$
top	$\top$	$\Delta^{\mathcal{I}}$
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
disjunction	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
universal quant.	$\forall R.C$	$\{d_1 \mid \forall d_2 \in \Delta^{\mathcal{I}}.(R^{\mathcal{I}}(d_1, d_2) \rightarrow d_2 \in C^{\mathcal{I}})\}$
existential quant.	$\exists R.C$	$\{d_1 \mid \exists d_2 \in \Delta^{\mathcal{I}}.(R^{\mathcal{I}}(d_1, d_2) \wedge d_2 \in C^{\mathcal{I}})\}$
role name	R	$R^{\mathcal{I}}$
role conjunction	$R_1 \sqcap R_2$	$R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}}$

Figure 3: Operators of $\mathcal{ALCR}$.

$$\begin{array}{rll}
\mathcal{I} \models C_1 \sqsubseteq C_2 & \text{iff} & C_1^{\mathcal{I}} \subseteq C_2{}^{\mathcal{I}} \\
\mathcal{I} \models R_1 \sqsubseteq R_2 & \text{iff} & R_1^{\mathcal{I}} \subseteq R_2{}^{\mathcal{I}} \\
\mathcal{I} \models a\!:\!C & \text{iff} & a^{\mathcal{I}} \in C^{\mathcal{I}} \\
\mathcal{I} \models (a,b)\!:\!R & \text{iff} & (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}.
\end{array}$$

Let $\Sigma = \langle T, A \rangle$ be a knowledge base and $\mathcal{I}$ an interpretation, then $\mathcal{I} \models \Sigma$ if for all $\phi \in T \cup A, \mathcal{I} \models \phi$. We say that Σ is *satisfiable* if there exists a model $\mathcal{I}$ such that $\mathcal{I} \models \Sigma$; it is *unsatisfiable* otherwise.

Let $\Sigma = \langle T, A \rangle$ be a knowledge base (with so-called simple, non-cyclic definitions, see [31]) in $\mathcal{ALCR}$. Σ can be transformed into an "unfolded" equivalent knowledge base $\Sigma' = \langle \emptyset, A' \rangle$ where all concept and role assertions are of the form $\bar{t}\!:\!\bigsqcap_i N_i$ [19]. Hence, from now on we will only consider knowledge bases with empty T-Boxes.

The definition of the normal form *nf* for DL assertions is a notational variation of Definition 3, obtained by exchanging $\vee$ by $\sqcup$, $\wedge$ by $\sqcap$, etc; and setting $nf(\bar{t}\!:\!N) = \bar{t}\!:\!nf(N)$. Again, for any assertion $\bar{t}\!:\!N$, *nf* always converges to a unique normal form.

Definition 12 (Clauses) A *clause* is a set Cl such that each element of Cl is either a concept assertion of the form $t\!:\!C$ or a role assertion of the form $(t_1, t_2)\!:\!R$, where $t, t_1, t_2 \in \mathsf{IND} \cup \mathsf{LAB}$. A formula in a clause is a *literal* if it is either a role assertion, a concept or negated concept assertion on an atomic concept, or a universal or negated universal concept assertion. The set S_Σ of clauses corresponding to a knowledge base $\Sigma = \langle \emptyset, A \rangle$ is the smallest set such that

- if $a\!:\!\bigsqcap_i C_i = nf(a\!:\!C)$ for $a\!:\!C \in A$ then $\{a\!:\!C_i\} \in S_\Sigma$,
- if $(a,b)\!:\!\bigsqcap_i R_i \in A$ then $\{(a,b)\!:\!R_i\} \in S_\Sigma$.

Formulas in a clause are simply assertions over an expanded set of individuals. Let Cl be a clause, and $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ a model over the expanded signature $\langle \mathsf{CON}, \mathsf{ROL}, \mathsf{IND} \cup \mathsf{LAB} \rangle$; we put $\mathcal{I} \models Cl$ if $\mathcal{I} \models \bigvee Cl$. A set of clauses S *has a model* if there is model $\mathcal{I}$ such that for all $Cl \in S$, $\mathcal{I} \models Cl$. Clearly, Σ is satisfiable iff S_Σ has a model.

Figure 4 shows the labeled resolution rules recast for the language $\mathcal{ALCR}$. The only differences with the rules in Figure 2 are that $(\sqcap)$ handles both concept and role conjunction, and that labels are now part of the language as the calculus directly manipulates concept and role assertions. Before proving soundness, completeness and termination we present a simple example of resolution in our system.

$$(\sqcap)\quad \frac{Cl \cup \{\bar{t}\!:\!N_1 \sqcap N_2\}}{\begin{array}{c}Cl \cup \{\bar{t}\!:\!N_1\}\\ Cl \cup \{\bar{t}\!:\!N_2\}\end{array}} \qquad (\neg\sqcap)\quad \frac{Cl \cup \{t\!:\!\neg(C_1 \sqcap C_2)\}}{Cl \cup \{t\!:\!\mathit{nf}(\neg C_1), t\!:\!\mathit{nf}(\neg C_2)\}}$$

$$(\mathrm{RES})\qquad \frac{Cl_1 \cup \{\bar{t}\!:\!N\} \quad Cl_2 \cup \{\bar{t}\!:\!\neg N\}}{Cl_1 \cup Cl_2}$$

$$(\forall)\qquad \frac{Cl_1 \cup \{t_1\!:\!\forall R.C\} \quad Cl_2 \cup \{(t_1,t_2)\!:\!R\}}{Cl_1 \cup Cl_2 \cup \{t_2\!:\!C\}}$$

$$(\neg\forall)\qquad \frac{Cl \cup \{t\!:\!\neg\forall R.C\}}{\begin{array}{c}Cl \cup \{(t,n)\!:\!R\}\\ Cl \cup \{n\!:\!\mathit{nf}(\neg C)\}\end{array}}, \text{ where } n \text{ is new.}$$

Figure 4: Labeled resolution rules for $\mathcal{ALCR}$.

Example 13 Consider the following description. Suppose that children of tall people are blond (1). Furthermore, all Tom's daughters are tall (2), but he has a non-blond grandchild (3). Can we infer that Tom has a son (4)?

(0) $\mathtt{FEMALE} \doteq \neg\mathtt{MALE}$
(1) $\mathtt{TALL} \sqsubseteq \forall\mathtt{Child.BLOND}$
(2) $\mathtt{tom}\!:\!\forall\mathtt{Child.}(\neg\mathtt{FEMALE} \sqcup \mathtt{TALL})$
(3) $\mathtt{tom}\!:\!\exists\mathtt{Child.}\exists\mathtt{Child.}\neg\mathtt{BLOND}$

---

(4) $\mathtt{tom}\!:\!\exists\mathtt{Child.MALE}$.

As is standard in DL, we use a new proposition letter `REST-TALL` to complete the partial definition in (1) to (1') $\mathtt{TALL} \doteq \forall\mathtt{Child.BLOND} \sqcap \mathtt{REST\text{-}TALL}$ (any partial definition $A \sqsubseteq B$ can be *completed* to an equivalent full definition $A = B \sqcap C$ for C a new concept, see [31]) and we resolve with the negation of the formula we want to infer (as a knowledge base Σ entails ϕ iff $\Sigma \cup \{\phi\}$ is inconsistent). After unfolding definitions (0) and (1') in (2) and applying *nf* we obtain the following three clauses:

1. $\{\mathtt{tom}\!:\!\forall\mathtt{Child.}\neg(\neg\mathtt{MALE} \sqcap \neg((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL}))\}$
2. $\{\mathtt{tom}\!:\!\neg\forall\mathtt{Child.}\forall\mathtt{Child.BLOND}\}$
3. $\{\mathtt{tom}\!:\!\forall\mathtt{Child.}\neg\mathtt{MALE}\}$.

Now we start resolving:

4. $\{\mathtt{s}\!:\!\neg\forall\mathtt{Child.BLOND}\}$ by $(\neg\forall)$ in 2
5. $\{(\mathtt{tom},\mathtt{s})\!:\!\mathtt{Child}\}$ by $(\neg\forall)$ in 2
6. $\{\mathtt{s}\!:\!\neg\mathtt{MALE}\}$ by $(\forall)$ in 3
7. $\{\mathtt{s}\!:\!\neg(\neg\mathtt{MALE} \sqcap \neg((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL})\}$ by $(\forall)$ in 1
8. $\{\mathtt{s}\!:\!\mathtt{MALE}, \mathtt{s}\!:\!((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL})\}$ by $(\neg\sqcap)$ in 7
9. $\{\mathtt{s}\!:\!((\forall\mathtt{Child.BLOND}) \sqcap \mathtt{REST\text{-}TALL})\}$ by (RES) in 6 and 8
10. $\{\mathtt{s}\!:\!\forall\mathtt{Child.BLOND}\}$ by $(\sqcap)$ in 9
11. $\{\mathtt{s}\!:\!\mathtt{REST\text{-}TALL}\}$ by $(\sqcap)$ in 9
12. $\{\}$ by (RES) in 4 and 10.

Thus, indeed, Tom has a son.

11

4.1 Soundness, Completeness and Termination

We will now prove that the resolution calculus for deciding knowledge base inconsistency in $\mathcal{ALCR}$ is sound and complete, and that a procedure for ensuring termination exists. The proofs can easily be adapted to prove completeness and termination for the resolution calculus presented in Section 3.

Given a set of clauses S_Σ corresponding to a knowledge base Σ, the notion of a refutation for S_Σ is as in Definition 9.

Theorem 14 (Soundness) The rules described in Figure 4 are sound. That is, if Σ is a knowledge base, then S_Σ has a refutation only if Σ is inconsistent.

Proof We prove that labeled resolution rules preserve satisfiability. We only discuss $(\neg\forall)$. Let $\mathcal{I}$ be a model of the premise. If $\mathcal{I}$ is a model of Cl we are done. If $\mathcal{I}$ is a model of $t:\neg\forall R.C$, then there exists d in the domain, such that $(t^{\mathcal{I}}, d) \in R^{\mathcal{I}}$ and $d \in \neg C^{\mathcal{I}}$. Let $\mathcal{I}'$ be like $\mathcal{I}$ except perhaps in the interpretation of n, where $n^{\mathcal{I}'} = d$. As n is new, $\mathcal{I}' \models t:\neg\forall R.C$. But now $\mathcal{I}' \models Cl \cup \{(t,n):R\}$ and $\mathcal{I}' \models Cl \cup \{n:\mathit{nf}(\neg C)\}$.

Next, we prove completeness. We follow the approach used in [20]: given a set of clauses S we aim to define a structure T_S such that

(†) if S is satisfiable, a model can be effectively constructed from T_S; and

(††) if S is unsatisfiable, a refutation can be effectively constructed from T_S.

In our case we also have to deal with A-Box information, that is, with named objects (concept assertions) and fixed constraints on relations (role assertions). We will proceed in stages. To begin, we will obtain a first structure to account for named states and their fixed relation constraints. After that we can use a simple generalization of results in [20]. We base our construction on trees which will help in guiding the construction of the corresponding refutation proof.

Let Σ be a knowledge base and S_Σ its corresponding set of clauses. In S_Σ we can identify a (possibly empty) subset of clauses RA of the form $\{(a,b):R\}$, and for each label a a (possibly empty) subset CA_a of clauses of the form $\{a:C\}$. Notice that S_Σ has no *mixed* clauses containing both concept and role assertions (in a single disjunction). Also, there are no disjunctive concept assertions on different labels, i.e., there is no clause Cl in S_Σ such that $Cl = Cl' \cup \{a:C_1\} \cup \{b:C_2\}$ for $a \neq b$. We will take advantage of these properties in the first steps of the completeness proof.

For each label a appearing in Σ, construct inductively a binary tree T_a whose nodes will be sets of clauses. Let the original tree T_a consist of the single node CA_a and repeat in alternation the following operations:

Operation A1. Repeat the following steps as long as possible:

- choose a leaf w. Replace in w any clause of the form $\{a:\neg(C_1 \sqcap C_2)\}$ by $\{a:\mathit{nf}(\neg C_1), a:\mathit{nf}(\neg C_2)\}$; and any clause of the form $\{a:C_1 \sqcap C_2\}$ by $\{a:C_1\}$ and $\{a:C_2\}$.

Operation A2. Repeat the following steps as long as possible:

- choose a leaf w and a clause Cl in w of the form $Cl = \{a:C_1, a:C_2\} \cup Cl'$;
- add two children w_1 and w_2 to w, where $w_1 = w \backslash \{Cl\} \cup \{\{a:C_1\}\}$ and $w_2 = w \backslash \{Cl\} \cup \{\{a:C_2\} \cup Cl'\}$.

12

The leaves of T_a give us the possibilities for "named states" in our model. We can view each leaf as a set S_a^j, representing a possible configuration for state a.

Proposition 15 Operation A (the combination of A1 and A2) terminates, and upon termination

1. all the leaves $S_a^1, \ldots, S_a^n$ of the tree are singleton sets of literal clauses,
2. if all $S_a^1, \ldots, S_a^n$ are refutable, then CA_a is refutable,
3. if one S_a^j is satisfiable, then CA_a is satisfiable.

Proof Termination is trivial. Item 1 holds by virtue of the construction, and 2 is proved by induction on the depth of the tree. We need only realize that by simple propositional resolution, if the two children of a node w are refutable, then so is w. Item 3 is also easy. Informally, Operation A "splits" disjunctions and "carries along" conjunctions. Hence if some S_a^j has a model we have a model satisfying all conjuncts in CA_a and at least one of each disjunct.

We now consider the set RA of role assertions. Let NAMES be the set of labels which appear in Σ. If a is in NAMES but CA_a is empty in S_Σ, define $S_a^1 = \{\{a:C, a:\neg C\}\}$ for some concept C. We define the set of sets of nodes $\mathcal{N} = \{N_i \mid N_i$ contains exactly one leaf of each $T_a\}$; each N_i is a possible set of constraints for the named worlds in a model of S_Σ.

For all i, we will now extend each set in N_i with further constraints. For each $S_a \in N_i$, start with a node w_a labeled by S_a.

Operation B1. Equal to *Operation A1.*

Operation B2. Repeat the following steps as long as possible:

- choose nodes w_a, w_b such that $\{(a,b):R\}$ in RA, $\{a:\forall R_i.C_i\} \in w_a$, $\{b:C_i\} \notin w_b$, where w_b is without children;
- add a child to w_b, $w_b' = w_b \cup \{\{b:C_i\}\}$.

Let N_i^* be the set of all leaves obtained from the forest constructed by applying Operation B (the combination of B1 and B2).

Proposition 16 Operation B terminates, and upon termination

1. all nodes created are derivable from $\bigcup N_i \cup RA$, and hence if a leaf is refutable so is $\bigcup N_i \cup RA$, and hence S_Σ too.
2. if some $\bigcup N_i^*$ is satisfiable, then S_Σ is satisfiable.

Proof To prove termination, notice that in each cycle the quantifier depth of the formulas considered decreases. Furthermore, it is not possible to apply twice the operation to nodes named by a and b and a formula $a:\forall R_i.C_i$.

As to item 1, each node is created by an application of the $(\forall)$ rule to members of $N_i \cup RA$ or clauses previously derived by such applications. Hence either already the original $\bigcup N_i^*$ is refutable and we can build a refutation for S_Σ as indicated in Proposition 15, or $\{\}$ can be derived from $\bigcup N_i$ by simple application of $(\forall)$.

To prove item 2, let $\mathcal{I}$ be a model of N_i^*. Define $\mathcal{I}' = \langle \Delta', \cdot^{\mathcal{I}'} \rangle$ as $\Delta' = \Delta$, $a^{\mathcal{I}'} = a^{\mathcal{I}}$ for all labels a, $C^{\mathcal{I}'} = C^{\mathcal{I}}$ for all atomic concepts C, and $R^{\mathcal{I}'} = R^{\mathcal{I}} \cup \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid \{(a,b){:}R\} \in RA\}$.

Observe that $\mathcal{I}'$ differs from $\mathcal{I}$ only in an extended interpretation of role symbols. By definition, $\mathcal{I}' \models RA$. It remains to prove that $\mathcal{I}' \models CA$. By Proposition 15, we are done if we prove that $\mathcal{I}' \models \bigcup N_i^*$. Since we only expanded the interpretation of relations, $\mathcal{I}$ and $\mathcal{I}'$ can only disagree on universal concepts of the form $a{:}\forall R.C$. By induction on the quantifier depth we prove this to be false.

Assume that $\mathcal{I}$ and $\mathcal{I}'$ agree on all formulas of quantifier depth less than n, and let $a{:}\forall R.C$ be of quantifier depth n, for $\{a{:}\forall R.C\} \in S_a^*$. Suppose $\mathcal{I}' \not\models \forall R.C$. This holds iff there exists b such that $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in R^{\mathcal{I}'}$ and $\mathcal{I}' \not\models b : C$. By the inductive hypothesis, $\mathcal{I} \not\models b{:}C$. Now, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ we are done. Otherwise, by definition $\{(a,b){:}R\} \in RA$. But then $\{b{:}C\} \in S_b^*$ by construction and as $\mathcal{I} \models S_b^*$, we also have $\mathcal{I} \models b{:}C$ — a contradiction.

Each N_i^* represents the "named core" of a model of S. The final step is to define the non-named part of the model. The following operations are performed to each set in each of the N_i^*, obtaining in such a way a forest F_i.

Fix N_i^*, and a. We construct a tree "hanging" from the corresponding $S_a^* \in N_i^*$. The condition that each node of the tree is named by either an individual or a new label (that is, all the formulas in a node have the same label) will be preserved as an invariant during the construction. Set the original tree u to S_a^* and repeat the following operations C1, C2 and C3 in succession until the end-condition holds.

Operation C1. Equal to *Operation A1.*

Operation C2. Equal to *Operation A2.*

Operation C3. For each leaf w of u,

- if for some concept we have $\{C\}, \{\neg C\} \in w$, do nothing;
- otherwise, since w is a set of unit clauses, we can write $w = \{\{t{:}C_1\}, \ldots, \{t{:}C_m\}, \{t{:}\forall R_{k_1}.A_1\}, \ldots, \{t{:}\forall R_{k_n}.A_n\}, \{t{:}\neg\forall R_{l_1}.P_1\}, \ldots, \{t{:}\neg\forall R_{l_q}.P_q\}\}$. Form the sets $w_i = \{\{nf(t'{:}\neg P_i)\}\} \cup S_i$, where t' is a new label, and $S_i = \{\{t'{:}A_h\} \mid \{t{:}\forall R_i.A_h\} \in w\}$, and append each of them to w as children marking the edges as R_i links. The nodes w_i are called the *projections* of w.

End-condition. Operation C3 is inapplicable.

Proposition 17 Operation C (the combination of C1, C2 and C3) cannot be applied indefinitely.

Definition 18 We call nodes to which Operation C1 or C2 has been applied of type 1, and those to which Operation C3 has been applied of type 2. The set of *closed nodes* is recursively defined as follows,

- if for some concept $\{t{:}C\}, \{t{:}\neg C\}$ are in w then w is closed,
- if w is of type 1 and all its children are closed, w is closed,

14

- if w is of type 2 and some of its children is closed, w is closed.

Let F_i be a forest that is obtained by applying Operations C1, C2, and C3 to N_i^* as often as possible. Then F_i is *closed* if any of its roots is closed.

Lemma 19 If one of the forests F_i is not closed, then S_Σ has a model.

Proof Let F_i be a non-closed forest. By a simple generalization of the results in [20, Lemma 2.7] we can obtain a model $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ of all roots S_a^* in F_i, from the trees "hanging" from them, i.e., a model of $\bigcup N_i^*$. By Proposition 16, S_Σ has a model.

Lemma 19 establishes the property (†) we wanted in our structure T_S. To establish (††) we need a further auxiliary result.

Proposition 20 Let w be a node of type 2. If one of its projections w_i is refutable, then so is w.

Proof Let w be a set of unit clauses $w = \{\{t : C_1\}, \ldots, \{t : C_m\}, \{t : \forall R_{k_1}.A_1\}, \ldots, \{t : \forall R_{k_n}.A_n\}, \{t : \neg\forall R_{l_1}.P_1\}, \ldots, \{t : \neg\forall R_{l_q}.P_q\}\}$. And let w_i be its refutable projection: $w_i = \{\{nf(t' : \neg P_i)\}\} \cup S_i$, where t' is a new label, and $S_i = \{\{t' : A_h\} \mid \{t : \forall R_i.A_h\} \in w\}$. We use resolution on w to arrive at the clauses in w_i from which the refutation can be carried out: Apply $(\neg\forall)$ to $\{t : \neg\forall R_i.P_i\}$ in w to obtain $\{t' : nf(t' : \neg P_i)\}$ and $\{(t, t') : R_i\}$. Now apply $(\forall)$ to all the clauses $\{t : \forall R_i.A_h\}$ in w to obtain $\{t' : A_h\}$.

Lemma 21 In a forest F_i, every closed node is refutable.

Proof For w a node in F_i, let $d(w)$ be the largest distance from w to a leaf.

If $d(w) = 0$, then w is a leaf, thus for some concept C, $\{t : C\}$ and $\{t : \neg C\}$ are in w. Using (RES) we immediately derive $\{\}$.

For the induction step, suppose the lemma holds for all w' such that $d(w') < n$ and that $d(w) = n$. If w is of type 1, let $w_1 = w \setminus \{Cl\} \cup \{Cl_1\}$ and $w_2 = w \setminus \{Cl\} \cup \{Cl_2\}$ be its children. By the inductive hypothesis there is a refutation for w_1 and w_2. By propositional resolution there is a refutation of w: repeat the refutation proof for w_2 but starting with w, instead of the empty clause we should obtain a derivation of Cl_2; now use the refutation of w_2. Suppose w is of type 2. Because w is closed, one of its projections is closed. Hence, by the inductive hypothesis it has a refutation. By Proposition 20, w itself has a refutation.

Theorem 22 (Completeness) The resolution method described above is complete: if Σ is a knowledge base, then S_Σ is refutable whenever Σ is inconsistent.

Proof We only need to put together the previous pieces. If Σ does not have a model then neither does S_Σ. By Lemma 19 all the forests F_i obtained from S_Σ are closed, and by Lemma 21, for each N_i^*, one of the sets $S_{a_j}^*$ is refutable. By Proposition 16, for all i, $\bigcup N_i \cup RA$ is refutable, and hence S_Σ is refutable.

Because we have shown how to *effectively* obtain a refutation from an unsatisfiable set of clauses we have also established termination. Notice that during the completeness proof we have used a *specific strategy* in the application of the resolution rules (crucially, the $(\neg\forall)$ rule is never applied twice to the same formula). By means of this strategy, we can guarantee termination of labeled modal resolution when verifying the consistency of any knowledge base in $\mathcal{ALCR}$.

Theorem 23 (Termination) Labeled resolution can effectively decide consistency of simple, acyclic knowledge bases in $\mathcal{ALCR}$.

We have now spelled out in detail our resolution method for the basic description logic $\mathcal{ALCR}$, and we could naturally consider extensions. For instance, in [14] some attention has been given to n-ary roles (in modal logic terms, n-ary modal operators). Our approach generalizes to this case without further problems.

An attractive idea which matches nicely with the resolution approach is to incorporate a limited kind of unification on "universal labels" of the form $x:C$, to account for on the fly unfolding of definitions and more general T-Boxes. The use of such universal labels would make it unnecessary to perform a complete unfolding of the knowledge base as a pre-processing step. The leitmotif would be "to do expansion by definitions only when needed in deduction." On the fly unfolding has already been implemented in tableaux based systems like KRIS [7].

5 Hybrid Logic

It is the turn of hybrid languages now. Of course, we have already been dealing with hybrid languages throughout the previous section: formulas in the A-Box are actually a restricted form of @ formulas. To handle the full basic hybrid language $\mathcal{H}(@)$, we need to provide rules for nominals and @. As before, we can get a hint of what is needed from the standard translation of this language into FO. The new clauses of ST are as follows:

$$\begin{array}{rcl|rcl} ST_x(i) & = & (x=i),\, i \in \mathsf{NOM} & ST_y(i) & = & (y=i),\, i \in \mathsf{NOM} \\ ST_x(@_i\phi) & = & \exists x.(x=i \wedge ST_x(\phi)) & ST_y(@_i\phi) & = & \exists y.(y=i \wedge ST_y(\phi)). \end{array}$$

Nominals and @ introduce a limited form of equational reasoning on labels. Indeed, a formula like $@_i j$ is true in a model iff i and j label the same state. Moreover, in the tableau treatment of hybrid languages, nominals and the @ operator have been considered as the mechanisms needed to formulate modal theories of state equality and state succession [10].

In the resolution tradition, Robinson and Wos [35] have introduced a rule called *paramodulation* to improve previous accounts of equational reasoning in FO:

$$\text{(PARAM)} \quad \frac{Cl_1 \cup \{t=s\} \qquad Cl_2 \cup \{\phi(u)\}}{(Cl_1 \cup Cl_2 \cup \{\phi(u/s)\})\sigma},$$

where σ is the most general unifier of t and u. In descriptions of paramodulation calculi one usually identifies the symmetric equations $s=t$ and $t=s$, includes (positive) factoring, and an inference rule that encodes resolution with the reflexivity axiom:

$$\text{(REF)} \quad \frac{Cl \cup \{t \neq s\}}{Cl\sigma},$$

where σ is the most general unifier of s and t. For a complete discussion of equational reasoning in first-order logics, we refer to [8]. We can introduce paramodulation in our direct resolution calculus for the basic hybrid language as follows:

$$\text{(PARAM)} \quad \frac{Cl_1 \cup \{@_t s\} \qquad Cl_2 \cup \{\phi(t)\}}{Cl_1 \cup Cl_2 \cup \{\phi(t/s)\}}$$

$$(\text{SYM})\quad \frac{Cl \cup \{@_t s\}}{Cl \cup \{@_s t\}} \qquad (\text{REF})\quad \frac{Cl \cup \{@_t \neg t\}}{Cl}$$

We need only one further rule to handle formulas of the form $@_s @_t \phi$ (see (@) below). Figure 5 shows the complete set.

$$(\wedge)\quad \frac{Cl \cup \{@_t(\phi_1 \wedge \phi_2)\}}{\begin{array}{c} Cl \cup \{@_t \phi_1\} \\ Cl \cup \{@_t \phi_2\} \end{array}} \qquad (\neg\wedge)\quad \frac{Cl \cup \{@_t \neg(\phi_1 \wedge \phi_2)\}}{Cl \cup \{@_t \mathit{nf}(\neg\phi_1), @_t \mathit{nf}(\neg\phi_2)\}}$$

$$(\text{RES})\quad \frac{Cl_1 \cup \{@_t \phi\} \quad Cl_2 \cup \{@_t \neg\phi\}}{Cl_1 \cup Cl_2}$$

$$([R])\quad \frac{Cl_1 \cup \{@_t [R]\phi\} \quad Cl_2 \cup \{@_t \neg[R]\neg s\}}{Cl_1 \cup Cl_2 \cup \{@_s \phi\}}$$

$$(\neg[R])\quad \frac{Cl \cup \{@_t \neg[R]\phi\}}{\begin{array}{c} Cl \cup \{@_t \neg[R]\neg n\} \\ Cl \cup \{@_n \mathit{nf}(\neg\phi)\} \end{array}}, \text{ where } n \text{ is new.}$$

$$(@)\quad \frac{Cl \cup \{@_t @_s \phi\}}{Cl \cup \{@_s \phi\}}$$

$$(\text{PARAM})\quad \frac{Cl_1 \cup \{@_t s\} \quad Cl_2 \cup \{\phi(t)\}}{Cl_1 \cup Cl_2 \cup \{\phi(t/s)\}}$$

$$(\text{SYM})\quad \frac{Cl \cup \{@_t s\}}{Cl \cup \{@_s t\}} \qquad (\text{REF})\quad \frac{Cl \cup \{@_t \neg t\}}{Cl}$$

Figure 5: Labeled resolution rules for $\mathcal{H}(@)$.

Two remarks on the rules above. Given that @ is self dual, we can define $\mathit{nf}(\neg @_t \phi) = @_t \mathit{nf}(\neg\phi)$; and for any formula ϕ in $\mathcal{H}(@)$, ϕ is satisfiable iff $@_t \phi$ is satisfiable, for a nominal t not appearing in ϕ. Hence, we can define the set S_ϕ of clauses corresponding to ϕ to be $\{\{@_t \mathit{nf}(\phi)\}\}$, where t does not appear in ϕ.

The soundness of the calculus is straightforward and the proof of Theorem 22 can be adapted in a way similar to the standard first-order case, to prove that paramodulation can handle @ and nominals (see [8]).

Theorem 24 The resolution calculus introduced in Figure 5 is sound and complete for $\mathcal{H}(@)$.

It is interesting to point out that optimizations and alternatives to paramodulation such as those discussed in [8] can now be investigated in the hybrid setting. In particular, we conjecture that heuristics can be devised to make the resolution calculus above terminating.

What about binders? Extending the system to account for hybrid sentences using $\downarrow$ is fairly straightforward. Consider the rule ($\downarrow$) below (again $\downarrow$ is self dual, so we don't need a rule for its negation):

$$(\downarrow) \quad \frac{Cl \cup \{@_t{\downarrow}x.\phi\}}{Cl \cup \{@_t\phi(x/t)\}}.$$

Notice that the rule transforms hybrid sentences into hybrid sentences. The full set of rules gives us a complete calculus for sentences in $\mathcal{H}(@, \downarrow)$. Let's go through a short example.

Example 25 We prove that ${\downarrow}x.\langle R\rangle(x \wedge p) \to p$ is a tautology. Consider the negation of the formula in clausal form

1. $\{@_i\underline{{\downarrow}x}.\neg[R]\neg(x \wedge p)\}$, $\{@_i\neg p\}$ by $(\downarrow)$
2. $\{@_i\underline{\neg[R]}\neg(i \wedge p)\}$, $\{@_i\neg p\}$ by $(\neg[R])$
3. $\{@_i\neg[R]\neg j\}$, $\{@_j(i\underline{\triangle}p)\}$, $\{i\colon\neg p\}$ by $(\wedge)$
4. $\{@_j\underline{i}\}$, $\{@_{\underline{j}}p\}$, $\{@_i\neg p\}$ by (PARAM)
5. $\{@_i\underline{p}\}$, $\{@_i\underline{\neg p}\}$ by (RES)
6. $\{\}$.

Of course, we cannot expect a heuristic ensuring termination of our calculus for $\mathcal{H}(@, \downarrow)$, as the satisfiability problem for $\mathcal{H}(\downarrow)$ is already undecidable (see [11]).

6 Conclusions

Blackburn [10] argues that hybrid languages can be used to internalize labeled deduction. Similar ideas play a fundamental role in the labeled resolution systems we introduced in this paper. Once again, nominals/labels together with the satisfiability operator : or @ are the key to achieving smooth and well behaved reasoning methods. The systems we introduced make clear that labeled resolution has many advantages in comparison with direct resolution proposals, thus supporting our claim that hybrid logic ideas can indeed be used to improve reasoning methods. We conclude the paper with a discussion of a number of independent directions for future research.

Once labels are introduced, the resolution method is very close to the tableaux approach, but we are still doing resolution. As we said, the rules $(\wedge)$, $(\neg\wedge)$ and $(\neg[R])$, prepare formulas to be fed into the resolution rules (RES) and $([R])$. And the aim is still to derive the empty clause instead of finding a model by exhausting a branch. But, is this method any better than tableaux? We don't think this is the correct question to ask. We believe that we learn different things from studying different methods. For example, Horrocks and Patel-Schneider [27] study a number of interesting optimizations of the tableaux implementation which were tested on the tableaux based theorem prover DLP. Some of their ideas can immediately be (or have already been) incorporated in our resolution method (lexical normalization and early detection of clashes, for instance), and others might be used in implementations of our method. At the same time, optimizations for direct resolution such as those discussed in [6] can also be exploited. For example, in implementations of the resolution algorithm, strategies for selecting the resolving pairs are critical. This kind of heuristics has been investigated by Auffray et al. [6] and some of their results easily extend to our framework. In certain cases, establishing completeness of these heuristics is even simpler because of our explicit use of resolution via nominals and @.

The issue of heuristics is very much connected to complexity. The basic heuristic used in the proof of Theorem 22 keeps the complete clause set "in memory" at all times and hence requires non-polynomial space. A similar situation occurs in clausal propositional resolution where the translation into clausal form can introduce an exponential blow up. We conjecture that a PSPACE heuristic for labeled resolution can be obtained by exploiting further the presence of labels (and given that we don't force a translation into full clausal form). Notice that nominals and @ let us keep track of the accessibility relation and we can define the notion of "being a member of a branch." Now we can attempt to use the tree property of modal languages to guide resolution. We used similar ideas in [5] to improve the performance of translation based resolution provers; see also [26] for translation based resolution methods which are able to polynomially simulate PSPACE tableaux.

The first author and Juan Heguiabehere are implementing a first prototype of the resolution method described in this paper. It would be interesting to perform empirical testing on the performance of this resolution prover along the lines of, for example [28], both in comparison with translation based resolution provers and those based on tableaux.

Finally, our completeness proof is constructive: if a refutation cannot be found, we can actually define a model for the formula or knowledge base. Hence, our methods can also be used for model extraction. How does this method perform in comparison with traditional model extraction from tableaux systems?

Acknowledgments

We want to thank Maarten Marx and an anonymous referee for useful comments and corrections. Carlos Areces was supported by grants from the Netherlands Organization for Scientific Research (NWO) under project number 612.069.006. Maarten de Rijke was supported by the Spinoza project 'Logic in Action' and by a grant from the Netherlands Organization for Scientific Research (NWO) under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, and 220-80-001.

References

[1] M. Abadi and Z. Manna. Nonclausal temporal deduction. *Lecture Notes in Computer Science*, 193:1–15, 1985.

[2] C. Areces. *Logic Engineering. The Case of Description and Hybrid Logics*. PhD thesis, Institute for Logic, Language and Computation, University of Amsterdam, Amsterdam, The Netherlands, October 2000.

[3] C. Areces and M. de Rijke. From description to hybrid logics, and back. In F. Wolter, H. Wansing, M. de Rijke, and M. Zakharyaschev, editors, *Advances in Modal Logic. Volume 3*. CSLI Publications, 2001.

[4] C. Areces, E. Franconi, R. Goré, M. de Rijke, and H. Schlingloff, editors. *Methods for Modalities 1*, volume 8(3). *Logic Journal of the IGPL*, 2000.

[5] C. Areces, J. Heguiabehere R. Gennari, and M. de Rijke. Tree-based heuristics in modal theorem proving. In *Proceedings of ECAI'2000*, pages 199–203, Berlin, Germany, 2000.

[6] Y. Auffray, P. Enjalbert, and J. Hebrard. Strategies for modal resolution: results and problems. *Journal of Automated Reasoning*, 6(1):1–38, 1990.

[7] F. Baader, B. Hollunder, B. Nebel, H. Profitlich, and E. Franconi. An empirical analysis of optimization techniques for terminological representation systems or: Making KRIS get a move on. *Journal of Applied Intelligence*, 4:109–132, 1994.

[8] L. Bachmair and H. Ganzinger. Equational reasoning in saturation-based theorem proving. In Bibel and Schmitt [9], chapter 11, pages 353–397.

[9] W. Bibel and P. Schmitt, editors. *Automated Deduction: A Basis for Applications*. Kluwer Academic Publisher, Dordrecht, 1998.

[10] P. Blackburn. Internalizing labelled deduction. *Journal of Logic and Computation*, 10(1):137–168, 2000.

[11] P. Blackburn and J. Seligman. Hybrid languages. *Journal of Logic, Language and Information*, 4(3):251–272, 1995. Special issue on decompositions of first-order logic.

[12] Bliksem Version 1.12. URL: `http://www.mpi-sb.mpg.de/~bliksem/`. Accessed Apr. 8, 2001.

[13] R. Brachman and J. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, 1985.

[14] D. Calvanese, G. De Giacomo, and M. Lenzerini. Conjunctive query containment in description logics with n-ary relations. In *Proceedings of DL-97*, pages 5–9, June 1997.

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

[16] H. de Nivelle. *Ordering Refinements of Resolution*. PhD thesis, Technische Universiteit Delft, Delft, The Netherlands, October 1994.

[17] H. de Nivelle, R. Schmidt, and U. Hustadt. Resolution-based methods for modal logics. In Areces et al. [4], pages 239–264.

[18] C. Dixon, M. Fisher, and M. Peim. Clausal temporal resolution. *ACM Transactions on Computational Logic*, 2(1), January 2001.

[19] F. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Deduction in concept languages: from subsumption to instance checking. *Journal of Logic and Computation*, 4(4):423–452, 1994.

[20] P. Enjalbert and L. Fariñas del Cerro. Modal resolution in clausal form. *Theoretical Computer Science*, 65(1):1–33, 1989.

[21] L. Fariñas del Cerro. A simple deduction method for modal logic. *Information Processing Letters*, 14:49–51, April 1982.

[22] C. Fermüller, A. Leitsch, T. Tammet, and N. Zamov. *Resolution Methods for the Decision Problem.* Springer-Verlag, Berlin, 1993. Lecture Notes in Artificial Intelligence.

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

[24] M. Fitting. Destructive modal resolution. *Journal of Logic and Computation*, 1(1):83–97, 1990.

[25] D. Gabbay. *Labelled Deductive Systems. Vol. 1.* The Clarendon Press Oxford University Press, New York, 1996. Oxford Science Publications.

[26] L. Georgieva, U. Hustadt, and R. A. Schmidt. Hyperresolution for guarded formulae. Submitted to *Journal of Symbolic Computation*, 2000.

[27] I. Horrocks and P. Patel-Schneider. Optimising description logic subsumption. *Journal of Logic and Computation*, 9(3):267–293, 1999.

[28] I. Horrocks, P. Patel-Schneider, and R. Sebastiani. An analysis of empirical testing for modal decision procedures. In Areces et al. [4], pages 293–324.

[29] U. Hustadt. *Resolution-Based Decision Procedures for Subclasses of First-Order Logic.* PhD thesis, Universität des Saarlandes, Saarbrücken, Germany, 1999.

[30] G. Mints. Resolution calculi for modal logics. *amst*, 143:1–14, 1989.

[31] B. Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43(2):235–249, 1990.

[32] H. Ohlbach. *A Resolution Calculus for Modal Logics.* PhD thesis, Universität Kaiserslautern, Germany, 1988.

[33] M. Paterson and M. Wegman. Linear unification. *Journal of Computer and System Sciences*, 16(2):158–167, 1978.

[34] A. Robinson and A. Voronkov, editors. *Handbook of Automated Reasoning.* Elsevier Science Publishers B. V., 2001.

[35] G. Robinson and L. Wos. Paramodulation and theorem-proving in first-order theories with equality. In *Machine Intelligence, 4*, pages 135–150. American Elsevier, New York, 1969.

[36] J. Robinson. A machine-oriented logic based on the resolution principle. *jacm*, 12(1):23–41, January 1965.

[37] R. Schmidt. *Optimised Modal Translation and Resolution.* PhD thesis, Universität des Saarlandes, Saarbrücken, Germany, 1997.

[38] SPASS Version 1.0.3. URL: `http://spass.mpi-sb.mpg.de/`. Accessed Apr. 8, 2001.

Sahlqvist Formulas in Hybrid Polyadic Modal Logics

Valentin Goranko[1]
Dimiter Vakarelov[2]

[1]Department of Mathematics, Rand Afrikaans University
PO Box 524, Auckland Park 2006, Johannesburg, South Africa
Email: `vfg@na.rau.ac.za`
[2]Faculty of Mathematics and Computer Science, Sofia University
blvd. James Bouchier 5, 1126 Sofia, Bulgaria
Email: `dvak@fmi.uni-sofia.bg`

Abstract

Building on a new approach to polyadic modal languages and Sahlqvist formulas introduced in [10] we define Sahlqvist formulas in *hybrid polyadic modal languages* containing nominals and universal modality or satisfaction operators. Particularly interesting is the case of *reversive* polyadic languages, closed under all 'inverses' of polyadic modalities because the minimal valuations arising in the computation of the first-order equivalents of polyadic Sahlqvist formulae are *definable* in such languages and that makes the proof of first-order definability and canonicity of these formulas a simple syntactic exercise. Furthermore, the first-order definability of Sahlqvist formulas immediately transfers to arbitrary polyadic languages, while the direct transfer of canonicity requires a more involved proof-theoretic analysis.

Keywords: hybrid polyadic modal logics, Sahlqvist formulas, nominals, universal modality, satisfaction operator, first-order definability, canonicity, completeness.

1 Introduction

In [10] we propose a new treatment of polyadic modal languages as '*purely modal polyadic languages*' where $\vee$ and $\wedge$ are treated as binary modalities (resp. a box and a diamond), boxes are composed as in PDL, and eventually every modal formula is represented as a polyadic box or a diamond. In these languages we defined a class of '*polyadic Sahlqvist formulas*' PSF which substantially extend the previously known class of Sahlqvist formulas (see [3]). The computation of the first-order equivalents of these formulas extends Sahlqvist-van Benthem's algorithm to an inductive procedure of computing the 'minimal' (first-order definable) valuations of the propositional variables in the formula.

Here we extend polyadic modal languages with nominals and universal modality or satisfaction operators, and define a large class of first-order definable and canonical formulas, extending PSF. In particular, we consider the case of *reversive* hybrid polyadic languages, closed under all 'inverses' of polyadic modalities. In these languages the minimal valuations arising in the computation of the first-order equivalents of polyadic Sahlqvist formulas are *definable*, which makes the proof of Sahlqvist theorem a simple syntactic exercise. As an immediate corollary we obtain first-order definability of Sahlqvist formulas in *arbitrary* polyadic languages, while the direct transfer of completeness apparently requires a more involved proof-theoretic analysis.

The structure of the paper: in the preliminary section we introduce purely modal polyadic languages and Sahlqvist formulas in them, as defined in [10], and in section 2 we extend these to the basic hybrid languages. The main result is in section 3 where we give an essentially syntactic proof of Sahlqvist theorem in reversive polyadic hybrid languages. The paper ends with a discussion on some general problems related to Sahlqvist formulas.

2 Preliminaries: Sahlqvist Formulas in Purely Modal Polyadic Languages

2.1 Purely Modal Polyadic Languages

Definition 1 ([10]) A *purely modal polyadic language* $\mathcal{L}_\tau$ contains propositional variables, negation $\neg$, and a *modal similarity type* τ consisting of a set of *basic modal terms* (modalities) with pre-assigned finite arities, including a 0-ary modality ι_0, a unary one ι_1, and a binary one ι_2. These distinguished modalities will be interpreted as follows: ι_0 by the constant $\top$ and its dual by $\bot$; ι_1 will be the self-dual identity; ι_2 will be $\vee$, and its dual — $\wedge$.

A *constant formula* will mean a formula containing no variables.

Definition 2 By simultaneous mutual induction we define the set of *modal terms* $MT(\tau)$ and their *arity function* ρ, and the set of *(purely) modal formulas* $MF(\tau)$ as follows:

(MT i) Every basic modal term is a modal term of the predefined arity.

(MT ii) Every constant formula is a 0-ary modal term.

(MT iii) If $n > 0$, α, $\beta_1, \ldots, \beta_n$ are modal terms and $\rho(\alpha) = n$, then $\alpha(\beta_1, \ldots, \beta_n)$ is a modal term and $\rho(\alpha(\beta_1, \ldots, \beta_n)) = \rho(\beta_1) + \ldots + \rho(\beta_n)$.

Modal terms of arity 0 will be called *modal constants.*

(MF i) Every propositional variable is a modal formula.

(MF ii) Every modal constant is a modal formula.

(MF iii) If A is a formula then $\neg A$ is a formula.

(MF iv) If $A_1, \ldots, A_n$ are formulas, α is a modal term and $\rho(\alpha) = n > 0$, then $[\alpha](A_1, \ldots, A_n)$ is a modal formula.

2

Note that constant formulas and 0-ary terms are regarded as both modal terms and formulas.

Example 3 Let α, β be basic modal terms such that $\rho(\alpha) = 1$, $\rho(\beta) = 2$. Then $\beta(\alpha, \iota_2)$ and $\beta(\alpha(\iota_0), \beta(\alpha, \iota_2))$ are 3-ary modal terms and $[\beta(\alpha, \iota_2)](\neg[\iota_1]q, [\alpha]p, \iota_0)$ is a modal formula. Furthermore, the modality $[\alpha]$ from the 2nd argument can be pulled into the external box resulting into an equivalent formula $[\beta(\alpha, \iota_2(\alpha, \iota_1))](\neg[\iota_1]q, p, \iota_0)$.

Some notation on formulas:

- $\langle\alpha\rangle(A_1, \ldots, A_n) = \neg[\alpha](\neg A_1, \ldots, \neg A_n)$;
- $\top = \iota_0, \bot = \neg\iota_0$;
- $A \vee B = [\iota_2](A, B), A \wedge B = \langle\iota_2\rangle(A, B)$,
  and respectively $A_1 \vee \ldots \vee A_n = [\iota_n](A_1, \ldots, A_n), A_1 \wedge \ldots \wedge A_n = \langle\iota_n\rangle(A_1, \ldots, A_n)$;
- $A \to B = \neg A \vee B$.

Positive and negative occurrences of variables and *positive and negative formulas* are defined as usual.

The *semantics* of purely modal languages is a straightforward combination of the standard Kripke semantics for polyadic modal languages and PDL-type of polymodal languages, taking into account the fact that conjunctions and disjunctions are now treated as modalities. In particular, a τ*-frame* is a structure $\langle W, \{R_\alpha\}_{\alpha \in MT(\tau)}\rangle$ where $R_\alpha \subseteq W^{\rho(\alpha)+1}$ is defined recursively by:

- $R_{\iota_0} = W,\ R_{\iota_1} = \{(x, x) | x \in W\},\ R_{\iota_2} = \{(x, x, x) | x \in W\}$.
- $R_{\alpha(\beta_1,\ldots,\beta_n)} = \{(x, x_{11}, \ldots, x_{1b_1}, \ldots, x_{n1}, \ldots, x_{nb_n}) \subseteq W^{b_1+\ldots+b_n+1} |$
  $\exists y_1 \ldots y_n (R_\alpha x y_1 \ldots y_n \wedge \bigwedge_{i=1}^n R_{\beta_i} y_i x_{i1} \ldots x_{ib_i})\}$
  where $\rho(\beta_i) = b_i, i = 1, \ldots n$,

Note that $R_{\iota_n} = \{(x, \ldots, x) \in W^{n+1} | x \in W\}$.

Now, the *truth definition* of a formula at a point of a Kripke model extends the classical modal case with the clause:

- $M, x \models [\alpha](A_1, \ldots, A_n)$ iff $\forall y_1, \ldots, y_n (R_\alpha x y_1 \ldots y_n \to \bigvee_{i=1}^n M, y_i \models A_i)$.
  In particular, $M, x \models \alpha$ iff $R_\alpha x$ for any modal constant α.

It follows immediately from the definitions that $[\alpha(\beta_1, \ldots, \beta_n)](A_{11}, \ldots, A_{1b_1}, \ldots, A_{n1}, \ldots, A_{nb_n})$ is logically equivalent to $[\alpha]([\beta_1](A_{11}, \ldots, A_{1b_1}), \ldots, [\beta_n](A_{n1}, \ldots, A_{nb_n}))$, and likewise for the diamonds.

Example 4

$$\begin{aligned}
[\beta(\alpha, \iota_2(\alpha, \iota_1))](\neg[\iota_1]q, p, \iota_0) &\equiv [\beta]([\alpha]\neg[\iota_1]q, [\iota_2]([\alpha]p, [\iota_1]\iota_0)) \\
&\equiv [\beta]([\alpha]\neg q, [\iota_2]([\alpha]p, \iota_0)), \\
&\quad \text{also written as } [\beta]([\alpha]\neg q, [\alpha]p \vee \top)) \\
&\equiv [\beta]([\alpha]\neg q, \top)) \\
&\equiv \top.
\end{aligned}$$

3

Accordingly, the *standard translation* ST generalizes the one for monadic languages with the clauses:

- $ST(\sigma) = R_\sigma(x)$ for every modal constant σ;
- $ST([\alpha](A_1,\ldots,A_n) = \forall y_1,\ldots,y_n(R_\alpha x y_1 \ldots y_n \to \bigvee_{i=1}^n ST(A_i)(y_i/x))$.

All propositional logical connectives, as defined here, have their standard interpretation, so the purely modal polyadic languages are not really different from the traditional ones regarding expressiveness, but rather more convenient to use.

2.2 Polyadic Sahlqvist Formulas

Definition 5 *Essentially box formula (EBF) of a variable* p is a formula $A = [\alpha](p)$ or $A = [\alpha](p, A_1,\ldots,A_n)$ where $A_1,\ldots,A_n$ are negative formulas not containing p. The variable p in such a formula is called *the essential variable of* A, while all other variables are *inessential* in the formula.

In particular, p, being $[\iota]p$, is an EBF of p, too. Intuitively, every EBF in a PSF can be regarded as a *unary* box over its essential variable. Of course, the essential variable in an EBF need not be the first argument, but to keep the notation simple we will be putting it in first position whenever possible.

Definition 6 A set of essentially box formulas is:

- *independent*, if no essential variable in a formula from the set occurs as an inessential variable in any formula from the set.
- *separated*, if all EBFs have different essential variables;
- *strongly independent*, if it is independent and separated.

Definition 7 A *Simple Polyadic Sahlqvist formula (SPSF)* is any modal constant σ, or a formula $A = [\alpha](A_1,\ldots,A_n)$ where α is an n-ary modal term and each formula A_i is either positive, or a negation of an essentially box formula, and the set of essentially box formulas whose negations are amongst $A_1,\ldots,A_n$ is independent. A SPSF $A = [\alpha](A_1,\ldots,A_n)$ is *very simple* if it is a modal constant or all essentially box formulas whose negations are amongst $A_1,\ldots,A_n$ are variables.

It is proved in [10] that every polyadic Sahlqvist formula as defined in [3] is equivalent to a conjunction of SPSFs.

Definition 8 Let $\mathcal{S} = \{B_1,\ldots,B_n\}$ be a set of EBFs with essential variables respectively $\{p_1,\ldots,p_n\}$. A *dependency digraph* $G(\mathcal{S})$ of $\mathcal{S}$ is defined as follows: the vertices of $G(\mathcal{S})$ are the variables $\{p_1,\ldots,p_n\}$ and p_i sends an arc to p_j if p_i occurs as an inessential variable in a formula from $\mathcal{S}$ with an essential variable p_j.

A digraph is called *acyclic* if it does not contain oriented cycles.

Definition 9 *Polyadic Sahlqvist formula (PSF)* is any modal constant σ, or $A = [\alpha](A_1,\ldots,A_n)$ where α is an n-ary modal term and each formula A_i is either positive, or a negation of an essentially box formula, and the dependency digraph of the set of essential variables in A is acyclic.

Furthermore, the class of PSFs can be considered closed under conjunctions. The particular case when there are no arcs in the dependency digraph corresponds to the class of SPSFs.

2.3 Pre-processing of Polyadic Modal Formulas

Definition 10 $A \sim B$ means that A and B are valid in the same frames.

A polyadic modal formula can be pre-processed in search of its representation as a PSF, which can then be simplified further, while preserving the class of frames in which it is valid, as follows:

1. The formula can be presented in a purely modal polyadic language as a polyadic box. (Logical equivalences simplifying the formula can be applied in the process.)
2. Elimination of monotone variables: every variable in a formula that has only positive/negative occurrences in that formula can be replaced by $\bot/\top$.
3. Pulling outwards and composing boxes. Eventually the formula can be written as $A = [\alpha](A_1, \dots, A_n)$, where $A_1, \dots, A_n$ are negated boxes or variables.
4. Appropriate substitutions changing the polarity of a variable (i.e. $\neg p/p$) can be applied in order to obtain a PSF.
5. Finally, a PSF can be transformed to one with a *strongly independent* set of EBFs. This can be done by means of successive splittings of an essential variable common for two EBFs into two different variables, using

$$[\alpha](\neg[\beta_1](p, \dots), \neg[\beta_2](p, \dots), \dots, P(p, \bar{q}), \dots) \sim$$
$$[\alpha](\neg[\beta_1](p_1, \dots), \neg[\beta_2](p_2, \dots), \dots, P(p_1 \vee p_2, \bar{q}), \dots),$$

   where ... in the formulas above stands for any string of arguments and $\bar{q}$ is any tuple of variables different from p.

Example 11 Let $\rho(\alpha) = 3$, $\rho(\beta) = 2$, $\rho(\gamma) = 1$.

$$\begin{array}{rcl}
[\alpha]([\gamma]p, \neg[\iota_2](p, q), [\beta](\langle\gamma\rangle p, q)) & \sim & [\alpha(\gamma, \iota_1, \beta)](p, \neg[\iota_2](p, q), \langle\gamma\rangle p, q) \\
 & & \text{(substitution: } \neg p/p, \neg q/q) \\
 & \sim & [\alpha(\gamma, \iota_1, \beta)](\neg p, \neg[\iota_2](\neg p, \neg q), \langle\gamma\rangle\neg p, \neg q) \\
 & \sim & [\alpha(\gamma, \iota_1, \beta)](\neg p, \langle\iota_2\rangle(p, q), \neg[\gamma]p, \neg q) \\
 & \sim & [\alpha(\gamma, \iota_1, \beta)](\neg p_1, \langle\iota_2\rangle(p_1 \vee p_2, q), \neg[\gamma]p_2, \neg q).
\end{array}$$

Clearly, these transformations can be performed *deductively* in the minimal polyadic modal logic.

2.4 Canonical Form of Polyadic Sahlqvist Formulas

Every PSF can be transformed by means of pre-processing (and permuting of arguments) to one in *canonical* form:

$$[\alpha](\neg[\beta_1](p_1,\ldots),\ldots,\neg[\beta_n](p_n,\ldots),C_1,\ldots,C_k)$$

where $\{[\beta_1](p_1,\ldots),\ldots,[\beta_n](p_n,\ldots)\}$ is a strongly independent set of EBFs with different essential variables respectively $p_1,\ldots,p_n$, and $C_1,\ldots,C_k$ are positive formulas, such that $p_1,\ldots,p_n$ are all variables occurring in the formula. Furthermore, if any of $\beta_1,\ldots,\beta_n$ is ι_m then the formula can be further transformed into a conjunction of simpler PSFs:

$$[\alpha](\neg[\iota_m](p_1,\ldots,p_m),\ldots,\neg[\beta_n](p_n,\ldots),C_1,\ldots,C_k)\equiv$$
$$[\alpha](\neg p_1,\ldots,\neg[\beta_n](p_n,\ldots),C_1,\ldots,C_k)\wedge\ldots\wedge[\alpha](\neg p_m,\ldots,\neg[\beta_n](p_n,\ldots),C_1,\ldots,C_k).$$

In particular, every SPSF can be transformed into

$$[\alpha](\neg[\beta_1]p_1,\ldots,\neg[\beta_n]p_n,C_1,\ldots,C_k)$$

where $\beta_1,\ldots,\beta_n$ are unary modal terms, $p_1,\ldots,p_n$ are different propositional variables, and $C_1,\ldots,C_k$ are positive formulas not containing any other variables but $p_1,\ldots,p_n$.

3 Hybrid Polyadic Modal Languages and Sahlqvist Formulas

In this section we consider extensions of purely modal polyadic languages with the basic ingredients of hybrid languages, viz. *universal modality, nominals, and satisfaction operators*, and expand the class of polyadic Sahlqvist formulas to each of these extensions. The universal modality $[\upsilon]$ is interpreted in a Kripke model by the Cartesian square of its universe. Nominals are a special sort of variables which admit only valuations which assign to them *singleton* sets. The satisfaction operator $@_nA$ expresses the claim that the formula A is true at the state in which the nominal n is evaluated. We note that these extensions boost the expressiveness of the modal languages considerably and many frame conditions non-definable in the classical modal language, such as irreflexivity etc., become definable in hybrid extensions. See [9, 8, 4, 2, 1], and Ch.7.3 in [3] for detailed studies of hybrid monadic languages.

3.1 The Basic Normal Polyadic Modal Logic

Suppose a purely modal polyadic language $\mathcal{L}_\tau$ is fixed. The basic normal polyadic logic $\mathcal{K}_\tau$ of $\mathcal{L}_\tau$ is axiomatized as follows:

3.1.1 Axioms:

A0) Enough propositional axioms.

A1.1) The polyadic analogues of the axiom scheme K:

$[\alpha](q_1, \ldots, q_{k-1}, p_1 \to p_2, q_{k+1}, \ldots, q_n) \to$
$([\alpha](q_1, \ldots, q_{k-1}, p_1, q_{k+1}, \ldots, q_n) \to [\alpha](q_1, \ldots, q_{k-1}, p_2, q_{k+1}, \ldots, q_n))$.

A1.2) Composition axiom

$[\alpha(\beta_1, \ldots, \beta_n)](p_{11}, \ldots, p_{1b_1}, \ldots, p_{n1}, \ldots, p_{nb_n}) \leftrightarrow$
$[\alpha]([\beta_1](p_{11}, \ldots, p_{1b_1}), \ldots, [\beta_n](p_{n1}, \ldots, p_{nb_n}))$, where $\rho(\beta_i) = b_i, i = 1, \ldots n$.

A1.3 $[\iota_1]p \leftrightarrow p$.

3.1.2 Rules of Inference:

R0) Uniform substitution of formulas for variables *SUB*.

R1) Modus ponens *MP*.

R2) Necessitation NEC_k : If $\vdash A$ then $\vdash [\alpha](B_1, \ldots, B_{k-1}, A, B_{k+1}, \ldots, B_n)$ for any n-ary modal term α and formulas $A, B_1, \ldots, B_{k-1}, B_{k+1}, \ldots, B_n$.

Proposition 12 Each inference rule above preserves validity in a (general) frame.

Proposition 13 $\mathcal{K}_\tau$ is sound and complete.

Proof Generalization of the canonical model completeness proof for $\mathcal{K}$. See e.g. [3], also [13].

3.2 Adding Universal Modality

Given a purely modal polyadic language $\mathcal{L}_\tau$, we denote by $\mathcal{L}_\tau^\upsilon$ its extension with a unary *universal modality* $[\upsilon]$, the semantics of which in a τ-frame $\langle W, \{R_\alpha\}_{\alpha \in MT(\tau)} \rangle$ is given by $R_\upsilon = W^2$.

The sets of modal terms and formulas in $\mathcal{L}_\tau^\upsilon$ accordingly extend those of $\mathcal{L}_\tau$.

The definition of PSF in $\mathcal{L}_\tau^\upsilon$ remains essentially the same, but the universal modality allows for additional preprocessing, using the following equivalences:

$$[\alpha](\neg[\upsilon]A, \ldots) \equiv \neg[\upsilon]A \vee [\alpha](\bot, \ldots);$$

$$\begin{gathered}[\alpha](\neg[\beta](\neg[\upsilon]A, \ldots), \ldots) \equiv \\ [\alpha]([\upsilon]A \wedge \neg[\beta](\bot, \ldots), \ldots) \equiv \\ [\alpha]([\upsilon]A, \ldots) \wedge [\alpha](\neg[\beta](\bot, \ldots), \ldots).\end{gathered}$$

Thus, for instance, the formula $[\alpha](\neg[\beta](p, \neg[\upsilon]q), \neg[\gamma](q, \neg[\upsilon]p), P_1 \ldots, P_k)$, where P_1, ..., P_k are positive formulas, is not a PSF but can be transformed to the conjunction of PSFs

$$\begin{gathered}[\alpha]([\upsilon]q, [\upsilon]p, P_1 \ldots, P_k) \wedge \\ [\alpha](\neg[\beta](p, \bot), [\upsilon]p, P_1 \ldots, P_k) \wedge \\ [\alpha]([\upsilon]q, \neg[\gamma](q, \bot), P_1 \ldots, P_k) \wedge \\ [\alpha](\neg[\beta](p, \bot), \neg[\gamma](q, \bot), P_1 \ldots, P_k).\end{gathered}$$

The basic normal polyadic logic $\mathcal{K}_\tau^\upsilon$ for $\mathcal{L}_\tau^\upsilon$ extends $\mathcal{K}_\tau$ with the axioms for υ:

A2.1) The S5 axioms for the universal modality $[\upsilon]$.

A2.2) $[\upsilon]p \to [\alpha](q_1, \ldots, q_{k-1}, p, q_{k+1}, \ldots, q_n)$.

Note that the necessitation rule schema R2 can now be replaced by

R2υ) Necessitation $NEC_k : \vdash A$ implies $\vdash [\upsilon]A$.

Proposition 14 $\mathcal{K}_\tau^\upsilon$ is sound and complete.

Proof Straightforward combination of the canonical completeness proofs for $\mathcal{K}_\tau$ and the extension of $\mathcal{K}$ with $[\upsilon]$ (see e.g. [9]).

Remark 15

1. We should note that the canonical relation R_υ^c corresponding to υ is *not the universal relation*, but an equivalence relation, so to make the interpretation of υ in the canonical model standard we consider R_υ^c-generated submodels of the whole canonical model.
2. On the other hand, first-order definability of formulas from $\mathcal{L}_\tau^\upsilon$ and the other hybrid languages will always refer to the standard semantics under consideration.

3.3 Adding Nominals

We now extend the polyadic modal language $\mathcal{L}_\tau$ to $\mathcal{L}_\tau^n$ by adding *nominals* $c_1, c_2, \ldots$. Henceforth by 'variable' we will mean an ordinary propositional variable, not a nominal. The definition of formulas extends accordingly, adding the clause that every nominal is a formula, and extending the set of modal terms as follows.

Definition 16 A formula of $\mathcal{L}_\tau^n$ is *pure* if it does not contain propositional variables.

Now the definition of *modal terms* in $\mathcal{L}_\tau^n$ extends the basic one with the clause: Every pure formula is a 0-ary modal term, i.e. modal terms can be parametrized with pure formulas.

Further, the definition of a model now accounts for the restriction on the nominals: an $\mathcal{L}_\tau^n$*-model* is a structure $M = \langle W, \{R_\alpha\}_{\alpha \in MT(\tau)}, V \rangle$ where V is a valuation for the propositional variables and the nominals such that $V(c)$ for any nominal c is a singleton. To simplify notation we shall write $V(c) = x$ instead of $\{x\}$. Then:

$$M, x \models c \text{ iff } V(c) = x.$$

Finally, the standard translation ST extends by

$$ST(c_i) := (x = y_i),$$

where $y_1, y_2, \ldots$ is a string of reserved variables associated with the nominals $c_1, c_2, \ldots$.

Proposition 17 Every pure formula is first-order definable.

Proof The pure formula $A(c_1, \ldots, c_n)$, where $c_1, \ldots, c_n$ are all nominals occurring in A, determines the first-order condition $\forall x \forall y_1, \ldots, \forall y_n ST(A)$.

Remark 18 A *positive formula* in a language with nominals is a formula in which all occurrences of *variables* are positive, while nominals can occur negatively, too.

Definition 19 *Polyadic Sahlqvist formulas* in $\mathcal{L}_\tau^{\mathbf{n}}$ are defined as before, (bearing in mind the extended definitions of modal terms and positive formulas).

In order for nominals to work well in the language, we need an additional mechanism which allows references (access) to the state named by a nominal from anywhere in the model. Such a mechanism can be the universal modality, or the satisfaction operator discussed in the next section.

The extension of $\mathcal{L}_\tau$ with υ and nominals will be denoted by $\mathcal{L}_\tau^{\upsilon,\mathbf{n}}$.

The basic logic $\mathcal{K}_\tau^{\upsilon,\mathbf{n}}$ of $\mathcal{L}_\tau^{\upsilon,\mathbf{n}}$ extends $\mathcal{K}_\tau^{\upsilon}$ with the following axioms for nominals:

A3.1) $\langle\upsilon\rangle\, c$,

A3.2) $\langle\upsilon\rangle\,(c \wedge p) \rightarrow [\upsilon](c \rightarrow p)$,

and the additional *covering rule* of inference

R3) *COV*: If $\vdash [\alpha](B_1, \ldots, B_{k-1}, c \rightarrow A, B_{k+1}, \ldots, B_n)$ for some nominal c not occurring in

$[\alpha](B_1, \ldots, B_{k-1}, A, B_{k+1}, \ldots, B_n)$ then $\vdash [\alpha](B_1, \ldots, B_{k-1}, A, B_{k+1}, \ldots, B_n)$.

Note that the rule of uniform substitution in a language with nominals allows substitution of formulas for variables and nominals for nominals.

Remark 20

1. It is easy to see that *COV* is deductively equivalent to the following *infinitary version*:

   COV^∞: If $\vdash [\alpha](B_1, \ldots, B_{k-1}, c \rightarrow A, B_{k+1}, \ldots, B_n)$ for *every* nominal c, then $\vdash [\alpha](B_1, \ldots, B_{k-1}, A, B_{k+1}, \ldots, B_n)$.

2. It is sufficient to formulate *COV* only for *basic* modal terms α and then it can be proved derivable for all composite terms in a language with υ.

Proposition 21 $\mathcal{K}_\tau^{\upsilon,\mathbf{n}}$ is sound and complete.

Proof Again, straightforward combination of the completeness proofs for $\mathcal{K}_\tau$ and the basic modal logic with nominals (see e.g. [12] or [8]).

Remark 22 The rule *COV*, which was introduced in [12] and used there to axiomatize PDL with nominals, can be omitted from the basic logic $\mathcal{K}_\tau^{\upsilon,\mathbf{n}}$ but is needed for its extensions.

Definition 23 A *pure extension* of $\mathcal{K}_\tau^{\upsilon,\mathbf{n}}$ is an extension with pure axioms.

Definition 24 Given a logic L in $\mathcal{L}_\tau^{\upsilon,\mathbf{n}}$, a *named canonical model* for L is every submodel, consisting only of maximal L-consistent sets which are closed under COV^∞, of a $[\upsilon]$-generated submodel of the ordinary canonical model M_L for L. L is *canonical* if all axioms of L are valid in the underlying frame $F_L^{\mathbf{n}}$ of every named canonical model $M_L^{\mathbf{n}}$.

Note that in every named canonical model the interpretation of υ is standard. As shown for the monadic case in [8] (see also [3]), every named canonical model satisfies the truth lemma and L is complete with respect to the class of named canonical models. Thus, every canonical logic in $\mathcal{L}_\tau^{\upsilon,\mathbf{n}}$ is complete. Moreover, every maximal L-consistent set in a named canonical model $M_L^{\mathbf{n}}$, being closed under COV^∞, contains a nominal, which is therefore evaluated at the corresponding state of $M_L^{\mathbf{n}}$, i.e. it can be used as a 'name' labeling that state. Thus, a modal language with nominals can refer to each state in a named canonical model, which entails the following fact.

Proposition 25 Every pure extension of $\mathcal{K}_\tau^{\upsilon,\mathbf{n}}$ is canonical.

Proof Since L is closed under substitution of nominals for nominals, all substitution instances of each pure axiom are valid in every named canonical model, hence each pure axiom is valid in the underlying frame. For a more detailed proof for the monadic case see [8] or [3].

Remark 26 It is known (see [8]) that the modal language with υ and nominals has the same expressive power regarding frame definability as the language with *difference operator* D (see [7, 3]). Same applies for polyadic modal languages, and thus results about Sahlqvist formulas in $\mathcal{L}_\tau^{\upsilon,\mathbf{n}}$ can be transferred to the polyadic language $\mathcal{L}_\tau^D$ and supplement related results in [14] where Sahlqvist theorem has been proved for a class of Sahlqvist formulas in *versatile* polyadic languages with D, also allowing certain additional rules of inference resembling COV.

3.4 Satisfaction Operators

Blackburn and Seligman proposed in [4] the use of so called *satisfaction operator* @ in hybrid modal languages instead of the universal modality, in order to keep the language local and reduce its complexity. The satisfaction operator works together with nominals and has the following semantics:

$M, s \vDash @_c A$ iff $M, V(c) \vDash A$ where c is a nominal.

Thus, @ introduces the truth at a state of a model explicitly in the language.

The language extending $\mathcal{L}_\tau$ with nominals and @ will be denoted by $\mathcal{L}_\tau^{@,\mathbf{n}}$. Its basic normal logic $\mathcal{K}_\tau^{@,\mathbf{n}}$ is axiomatized over $\mathcal{K}_\tau$ by adding the following axioms, where c, d denote nominals (see [5, 1, 3] for the monadic case):

A@1 $@_c(p \to q) \to (@_c p \to @_c q)$,

A@2 $@_c p \leftrightarrow \neg @_c \neg p$,

A@3 $c \wedge p \to @_c p$,

A@4 $@_c c$,

A@5 $@_c d \leftrightarrow @_d c$,

A@6 $@_c @_d p \leftrightarrow @_d p$,

A@7 $\langle \alpha \rangle (\ldots, @_c p, \ldots) \to @_c p$,

and the following rules:

R@0 *@-GEN*: If $\vdash A$ then $\vdash @_c A$.

R@1 *PASTE*: If $\vdash @_d \langle\alpha\rangle(\ldots, c, \ldots) \wedge @_c B \to A$ for some nominal c distinct from d and not occurring in A and B then $\vdash @_d \langle\alpha\rangle(\ldots, B, \ldots) \to A$.

The rule *PASTE* can be rewritten by contraposition as follows:
If $\vdash A \wedge @_c B \to @_d[\alpha](\ldots, \neg c, \ldots)$ for some nominal c distinct from d and not occurring in A and B then $\vdash A \to @_d[\alpha](\ldots, \neg B, \ldots)$.
The following rule is easily derivable from *PASTE*, taking $\alpha = \iota_1$ and $B = d$:

R@2 *NAME*: If $\vdash @_c A$ for some nominal c not occurring in A then $\vdash A$.

Clearly, the rules *NAME* and *PASTE* are analogues of particular cases of *COV*, and it is interesting to note that they suffice.
Combining the completeness proofs for $\mathcal{K}_\tau$ and for the basic hybrid modal logic with nominals and @ (see [3]) we obtain:

Proposition 27 $\mathcal{K}_\tau^{@,\mathbf{n}}$ is sound and complete.

Remark 28 In a language with υ and nominals, @ is definable as follows:
$@_c A := \langle\upsilon\rangle(c \wedge A)$.

4 Sahlqvist Theorem in Reversive Languages with Nominals

In [10] we show that every polyadic Sahlqvist formula is first-order definable and canonical, and give an algorithm, extending Sahlqvist – van Benthem's algorithm, for computing their first-order equivalents. Here we will show that in sufficiently rich hybrid polyadic languages the first-order equivalents can be computed *deductively within the modal logic* and the completeness can be obtained as a corollary to Proposition 25.

4.1 Reversive Polyadic Languages

Definition 29 A purely modal polyadic language is *reversive* if together with every n-ary modal term α it contains its *inverses* $\alpha^{-1}, \ldots, \alpha^{-n}$, where for each $k = 1, \ldots, n$:

$$xR_{\alpha^{-k}} y_1 \ldots y_k \ldots y_n \text{ iff } y_k R_\alpha y_1 \ldots x \ldots y_n.$$

Remark 30

1. It *does not* suffice to require this condition only for the basic modal terms from the type τ, because reversiveness is apparently not preserved by compositions. For instance, the inverse δ^{-1} of $\delta = \gamma(\beta, \alpha)$ where α is a unary term and β, γ are binary terms seems not definable in terms of α, β, γ and their inverses.
2. Note that $(\alpha^{-k})^{-k} = \alpha$.

An arbitrary purely modal polyadic language $\mathcal{L}_\tau$ can be extended to a reversive one $\mathcal{L}_{\tau r}$ by adding the following clause to the definition of modal terms.

Definition 31 (MT iv) If $n > 0$, $k \leq n$ and α is an n-ary modal term then α^{-k} is an n-ary modal term, too.

The semantics extends accordingly:

$$\begin{array}{rcl} R_{\alpha^{-k}} & = & \{(x_0, x_1, \ldots, x_{k-1}, x_k, x_{k+1}, \ldots, x_n) \subseteq W^{n+1} | \\ & & (x_k, x_1, \ldots, x_{k-1}, x_0, x_{k+1}, \ldots, x_n) \in R_\alpha\}. \end{array}$$

Thus, $R_{\alpha^{-k}}$ is obtained from R_α by transposing the 0-th and the k-th arguments. In particular, for a unary term α, $R_{\alpha^{-1}}$ is the usual inverse of α, as expected.

As we will see further, reversiveness provides the language with the degree of expressiveness necessary for the explicit definition of the minimal valuations of the variables, used in the computation of the first-order equivalents of PSFs.

4.2 Reversive vs. Versatile Languages

In [14] Venema introduces *versatile* polyadic languages, in which the set of modal terms is closed under *cyclic* permutations of the arguments of the corresponding relations, i.e. with every n-ary modal term α a versatile language contains its *conjugates* $\alpha^{(1)}, \ldots, \alpha^{(n)}$, where for each $k = 1, \ldots, n$:

$$xR_{\alpha^{(k)}} y_1 \ldots y_k \ldots y_n \text{ iff } y_k R_\alpha y_{k+1} \ldots y_n, x, y_1, \ldots, y_{k-1};$$

in particular $xR_{\alpha^{(n)}} y_1 \ldots y_k \ldots y_n$ iff $y_n R_\alpha x, y_1, \ldots, y_{n-1}$.

Ostensibly, versatile languages are weaker than reversive ones since only the modalities corresponding to cyclic permutations of the relational arguments can be defined in them, while in a reversive language by composing α and $\alpha^{-1}, \ldots, \alpha^{-n}$ one can construct all modal terms whose relations are obtained by permutations of the arguments of R_α.

Still, the difference in expressiveness between versatile and reversive languages is only apparent: every inverse term α^{-k} to an n-ary term α can be obtained from a conjugate of α by permuting appropriately the arguments of the box, and hence both types of languages have *the same* expressive power, and moreover are interdefinable. Indeed,

$$xR_{\alpha^{-k}} y_1 \ldots y_k \ldots y_n \text{ iff } y_k R_\alpha y_1 \ldots y_{k-1} x y_{k+1} \ldots y_n \text{ iff } xR_{\alpha^{(k)}} y_{k+1} \ldots y_n y_1 \ldots y_{k-1}.$$

Therefore, $[\alpha^{-k}](A_1, \ldots, A_k \ldots A_n) = [\alpha^{(k)}](A_{k+1}, \ldots, A_n, A_1, \ldots, A_{k-1})$.

Hence, the difference between the two types is only technical and not essential. What are the pros and cons of preferring one to another? On one hand, it is easy to see that versatility of polyadic languages is preserved under compositions, so it only needs to be required for the basic modal terms, while this is apparently not the case for reversive languages. On the other hand, working with versatile languages would require either extending the language with operators permutating the arguments of boxes, or keeping track of these permutations, which is technically inconvenient. That is why we choose to work with reversive languages hereafter, but all that follows holds accordingly in versatile languages, too.

4.3 The Minimal Logics of Reversive Polyadic Languages

The minimal normal modal logic $\mathcal{K}_{\tau r}$ of a reversive polyadic language $\mathcal{L}_{\tau r}$ is axiomatized over $\mathcal{K}_{\tau r}$ by adding the following *axiom schemes for inverse modalities*:

A4.1) $A \to [\alpha](\neg B_1, \ldots, \neg B_{k-1}, \langle \alpha^{-k} \rangle (B_1, \ldots, B_{k-1}, A, B_{k+1}, \ldots, B_n), \neg B_{k+1}, \ldots, \neg B_n)$,

A4.2) $[(\alpha^{-k})^{-k}](\ldots, A, \ldots) \leftrightarrow [\alpha](\ldots, A, \ldots)$.

In particular, in a tense language axiom A4.1 becomes $A \to [\alpha] \langle \alpha^{-1} \rangle A$.

Likewise, the schemes above added to $\mathcal{K}^{\upsilon}_{\tau}$ and $\mathcal{K}^{\upsilon,\mathbf{n}}_{\tau}$ produce the minimal logics $\mathcal{K}^{\upsilon}_{\tau r}$ and $\mathcal{K}^{\upsilon,\mathbf{n}}_{\tau r}$ in the respective reversive languages with υ and nominals..

Remark 32 1. The following are easily derivable:

(a) $[\upsilon^{-1}]p \leftrightarrow [\upsilon]p$,

(b) $[\iota_1^{-1}]p \leftrightarrow p$,

(c) $[\iota_2^{-i}](p, q) \leftrightarrow [\iota_2](p, q)$, $i = 1, 2$.

2. Axiom A4.2 can be replaced by the dual of A4.1: $A \to [\alpha^{-k}](\neg B_1, \ldots, \neg B_{k-1}, \langle \alpha \rangle (B_1, \ldots, B_{k-1}, A, B_{k+1}, \ldots, B_n), \neg B_{k+1}, \ldots, \neg B_n)$.

   These two are inter-derivable using A4.1.

3. In a reversive language with universal modality, the scheme A4.1 can be replaced by:

   A4.1.υ) $[\upsilon](A \to [\alpha](B_1, \ldots, B_{k-1}, B, B_{k+1}, \ldots, B_n)) \leftrightarrow$
   $[\upsilon](\neg B \to [\alpha^{-k}](B_1, \ldots, B_{k-1}, \neg A, B_{k+1}, \ldots, B_n))$.

Theorem 33 Each of $\mathcal{K}_{\tau r}$, $\mathcal{K}^{\upsilon}_{\tau r}$ and $\mathcal{K}^{\upsilon,\mathbf{n}}_{\tau r}$ is sound and complete.

Definition 34 Formulas A and B of a modal language $\mathcal{L}$ are *axiomatically equivalent* if each of them is derivable in the logic axiomatized with the other over the basic normal logic for $\mathcal{L}$.

Proposition 35 Axiomatically equivalent formulas define the same classes of frames.

Proof Immediate from Proposition 12.

4.4 Computing First-Order Equivalents of Sahlqvist Formulas

Here we show that every PSF in $\mathcal{L}^{\upsilon,\mathbf{n}}_{\tau r}$ is axiomatically equivalent in $\mathcal{K}^{\upsilon,\mathbf{n}}_{\tau r}$ to a pure formula, which is obtained by systematic computation and substitution of the minimal valuations for the essential variables. This can be done deductively within $\mathcal{K}^{\upsilon,\mathbf{n}}_{\tau r}$ because the minimal valuations here are definable by means of pure formulas. Thus we obtain a *modal calculus of Sahlqvist equivalents* which computes the first-order Sahlqvist equivalents of PSFs.

Hereafter in this section $\vdash A$ will refer to derivability in the basic logic of the minimal language containing the formula A.

Lemma 36 (Monotonicity lemma) Let $\bar{p} = p^1, \dots p^m$ be a list of positive occurrences of a variable p in a formula A and B, C be any formulas. Denote by $A(Q/\bar{p})$ the result of the uniform substitution of a formula Q for the occurrences $\bar{p}$ in A.

1. If $\vdash B \to C$ then $\vdash A(B/\bar{p}) \to A(C/\bar{p})$.
2. In a language with $v : \vdash [v](B \to C) \to (A(B/\bar{p}) \to A(C/\bar{p}))$.
3. In a language with $@ : \vdash @_c B \to (A(c/\bar{p}) \to A(B/\bar{p}))$.

Proof Easy structural induction on A, using the axioms for the universal modality, (respectively for @).

Lemma 37 The formula $A = [\alpha](\neg[\gamma](p, \neg Q_1, \dots, \neg Q_n), P_1, \dots, P_k)$, where $P_1, \dots, P_k$ are formulas positive in p, is axiomatically equivalent over $\mathcal{K}_{\tau r}$ to $A^q = [\alpha](\neg q, P_1^q, \dots, P_k^q)$ where $P_i^q = P_i(\langle\gamma^{-1}\rangle (q, Q_1, \dots, Q_n)/p)$, for any variable q not occurring in A and $i = 1, \dots, k$.

Proof First, suppose $\vdash A$ and substitute $\langle\gamma^{-1}\rangle (q, Q_1, \dots, Q_n)$ for p in A. Then

$$\vdash [\alpha](\neg[\gamma](\langle\gamma^{-1}\rangle (q, Q_1, \dots, Q_n), \neg Q_1, \dots, \neg Q_n), P_1^q, \dots, P_k^q). \qquad (*)$$

From axiom A4.1 we obtain by contraposition

$$\vdash \neg[\gamma](\langle\gamma^{-1}\rangle (q, Q_1, \dots, Q_n), \neg Q_1 \dots, \neg Q_n) \to \neg q.$$

Now, from $(*)$ by the monotonicity Lemma 36 we get $\vdash [\alpha](\neg q, P_1^q, \dots, P_k^q)$.

Conversely, suppose $\vdash [\alpha](\neg q, P_1^q, \dots, P_k^q)$. Let $Q = \langle\gamma^{-1}\rangle ([\gamma](p, \neg Q_1, \dots, \neg Q_n), Q_1, \dots, Q_n)$. Then, substituting $[\gamma](p, \neg Q_1, \dots, \neg Q_n)$ for q we get

$$\vdash [\alpha](\neg[\gamma](p, \neg Q_1, \dots, \neg Q_n), P_1(Q/p), \dots, P_k(Q/p)). \qquad (**)$$

From axioms A4.1 and A4.2, we obtain by contraposition

$$\vdash \langle\gamma^{-1}\rangle ([\gamma](p, \neg Q_1, \dots, \neg Q_n), Q_1, \dots, Q_n) \to p,$$

whence by the monotonicity Lemma 36

$$\begin{aligned} \vdash\ & [\alpha](\neg[\gamma](p, \neg Q_1, \dots, \neg Q_n), P_1(Q/p), \dots, P_k(Q/p)) \\ & \to [\alpha](\neg[\gamma](p, \neg Q_1, \dots, \neg Q_n), P_1, \dots, P_k), \end{aligned}$$

hence $\vdash [\alpha](\neg[\gamma](p, \neg Q_1, \dots, \neg Q_n), P_1, \dots, P_k)$ by (**).

Likewise the lemma applies to any $A = [\alpha](P_1, \dots, \neg[\gamma](p, \neg Q_1, \dots, \neg Q_n), \dots, P_k)$.

Theorem 38 Every PSF is axiomatically equivalent in $\mathcal{K}_{\tau r}$ to a very simple PSF.

Proof Let $A = [\alpha](\neg B_1, \dots, \neg B_n, C_1, \dots, C_k)$ be a pre-processed PSF with EBFs $B_1, \dots, B_n$ and different essential variables resp. $q_1, \dots, q_n$. Let the dependency digraph of A determine a precedence order $\prec$ on these variables. We transform A into A° by a sequence of intermediate formulas $A = A_1, \dots, A_s = A^\circ$ obtained by

14

successive replacement of all EBFs by variables one by one inductively on $\prec$, by applying Lemma 37. When there are more than one EBFs to be eliminated at a time, they are dealt with one by one in arbitrary order. Note that the definition of $\prec$ ensures that at every step the condition requiring the essential variable of the EBF which is being eliminated to be positive in all other arguments, needed for the application of Lemma 37, will be maintained.

The resulting formula A° is $[\alpha](\neg q_1, \ldots, \neg q_n, D_1, \ldots, D_k)$ where $D_1, \ldots, D_k$ are positive formulas.

Lemma 39 The formula $A = [\alpha](\neg q, P_1, \ldots, P_m)$ where $P_1, \ldots, P_m$ are formulas positive in q is axiomatically equivalent over either of $\mathcal{K}_\tau^{\upsilon,\mathbf{n}}$ or $\mathcal{K}_\tau^{@,\mathbf{n}}$ to a $\mathcal{L}_\tau^{\mathbf{n}}$-formula $A^c = [\alpha](\neg c, P_1^c, \ldots, P_m^c)$ where $P_i^c = P_i(c/q)$ for any nominal c not occurring in A and $i = 1, \ldots, m$.

Proof If $\vdash A$ then $\vdash A^c$ by substitution of c for q in A. Conversely, let $\vdash A^c$.

1. The derivation in $\mathcal{K}_\tau^{\upsilon,\mathbf{n}}$. First, we shall prove $\vdash [\alpha](c \to \neg q, P_1, \ldots, P_m)$. It is easy to see that $\vdash [\upsilon](c \to A) \vee [\upsilon](c \to \neg A)$, so it suffices to prove

$$\vdash ([\upsilon](c \to q) \vee [\upsilon](c \to \neg q)) \to [\alpha](c \to \neg q, P_1, \ldots, P_m),$$

i.e.

$$\vdash [\upsilon](c \to q) \to [\alpha](c \to \neg q, P_1, \ldots, P_m) \text{ and} \qquad (i)$$
$$\vdash [\upsilon](c \to \neg q) \to [\alpha](c \to \neg q, P_1, \ldots, P_m). \qquad (ii)$$

For (i) first note that $\vdash [\alpha](\neg c, P_1^c, \ldots, P_m^c) \to [\alpha](c \to \neg q, P_1^c, \ldots, P_m^c)$ since $\vdash \neg c \to (c \to \neg q)$, by monotonicity. Thus, $\vdash [\alpha](c \to \neg q, P_1^c, \ldots, P_m^c)$. Then, again by monotonicity, $\vdash [\upsilon](c \to q) \to ([\alpha](c \to \neg q, P_1^c, \ldots, P_m^c) \to [\alpha](c \to \neg q, P_1, \ldots, P_m))$, hence $[\upsilon](c \to q) \to [\alpha](c \to \neg q, P_1, \ldots, P_m)$.

(ii) is immediate from axiom A2.2.

Now, from $\vdash [\alpha](c \to \neg q, P_1, \ldots, P_m)$ applying *COV* we get $\vdash [\alpha](\neg q, P_1, \ldots, P_m)$.

2. The derivation in $\mathcal{K}_\tau^{@,\mathbf{n}}$. Let d be a nominal not occurring in $[\alpha](\neg c, P_1, \ldots, P_m)$. From $\vdash [\alpha](\neg c, P_1^c, \ldots, P_m^c)$ we get $\vdash @_d[\alpha](\neg c, P_1^c, \ldots, P_m^c)$ by *@-GEN*. Then, by the monotonicity lemma, $\vdash @_c q \to (@_d[\alpha](\neg c, P_1^c, \ldots, P_m^c) \to @_d[\alpha](\neg c, P_1, \ldots, P_m))$, so $\vdash @_c q \to @_d[\alpha](\neg c, P_1, \ldots, P_m)$. Now applying the contrapositive version of *PASTE* we get $\vdash @_d[\alpha](\neg q, P_1, \ldots, P_m)$, hence $\vdash [\alpha](\neg q, P_1, \ldots, P_m)$ by applying the rule *NAME*.

Lemma 40 Every very simple PSF $A = [\alpha](\neg q_1, \ldots, \neg q_n, Q_1, \ldots, Q_k)$ in $\mathcal{L}_\tau^{\upsilon,\mathbf{n}}$ or $\mathcal{L}_\tau^{@,\mathbf{n}}$ is axiomatically equivalent to the pure formula $A^{\mathbf{p}} = [\alpha](\neg c_1, \ldots, \neg c_n, P_1, \ldots, P_k)$, where $P_1, \ldots, P_k$ are obtained from $Q_1, \ldots, Q_k$ by substitutions $c_1/q_1, \ldots, c_n/q_n$ where $c_1, \ldots, c_n$ are fresh nominals.

Proof We can assume that no other variables but $q_1, \ldots, q_n$ occur in $Q_1, \ldots, Q_k$. (Otherwise they can be replaced by $\bot$). Applying several times Lemma 39 we consecutively replace the variables $q_1, \ldots, q_n$ by new nominals $c_1, \ldots, c_n$ and eventually obtain $A^{\mathbf{p}} = [\alpha](\neg c_1, \ldots, \neg c_n, P_1, \ldots, P_k)$.

Theorem 41 Every PSF $A = [\alpha](\neg B_1, \ldots, \neg B_n, Q_1, \ldots, Q_k)$ in $\mathcal{L}^{v,\mathbf{n}}_{\tau r}$ or $\mathcal{L}^{@,\mathbf{n}}_{\tau r}$ is axiomatically equivalent to a pure formula $A^{\mathbf{p}} = [\alpha](\neg c_1, \ldots, \neg c_n, P_1, \ldots, P_k)$ where $P_1, \ldots, P_k$ are obtained from $Q_1, \ldots, Q_k$ by appropriate pure substitutions.

Proof Immediate from Theorem 38 and Lemma 40.

Now, the fist-order equivalent of the pure formula $A^{\mathbf{p}}$ is obtained immediately. Let $d_1, \ldots, d_m$ be all nominals occurring in $P_1, \ldots, P_k$ and different from $c_1, \ldots, c_n$. Further, let $\bar{y} = y_1, \ldots, y_n$ be the variables used in the standard translation ST of the nominals $c_1, \ldots, c_n$, $\bar{x} = x_1, \ldots, x_m$ be the variables used for $d_1, \ldots, d_m$, and $x, z_1, \ldots, z_k$ be a string of variables disjoint from $\bar{y}$ and $\bar{x}$. Then

$$FO(A^{\mathbf{p}}) = \forall x \bar{x} \bar{y} \bar{z} \left(R_\alpha x y_1 \ldots y_n z_1 \ldots z_k \rightarrow \bigvee_{j=1}^{k} ST(P_j)(z_j/x) \right).$$

Example 42 The formula

$$A = [\alpha](\neg[\beta]p_1, \neg[\gamma](p_2, \neg[\delta](p_1, p_3)), \neg[\gamma](p_3, \neg\langle\delta\rangle(p_1, p_1)), [\beta]p_1 \wedge p_2)$$

has essential variables p_1, p_2, p_3 and the ordering on them induced by the dependency graph is: $p_1 \prec p_2$, $p_1 \prec p_3$, $p_3 \prec p_2$, so their essentially box formulas should be eliminated in order: p_1, p_3, p_2, applying Lemma 37. That elimination transforms the formula into a very simple PSF in 3 steps as follows:

$A_1 = [\alpha](\neg q_1, \neg[\gamma](p_2, \neg[\delta](\langle\beta^{-1}\rangle q_1, p_3)), \neg[\gamma](p_3, \neg\langle\delta\rangle(\langle\beta^{-1}\rangle q_1, \langle\beta^{-1}\rangle q_1)),$
$[\beta]\langle\beta^{-1}\rangle q_1 \wedge p_2),$

$A_2 = [\alpha](\neg q_1, \neg[\gamma](p_2, \neg[\delta](\langle\beta^{-1}\rangle q_1, \langle\gamma^{-1}\rangle(q_2, \langle\delta\rangle(\langle\beta^{-1}\rangle q_1, \langle\beta^{-1}\rangle q_1)))), \neg q_2,$
$[\beta]\langle\beta^{-1}\rangle q_1 \wedge p_2),$

$A^\circ = A_3 = [\alpha](\neg q_1, \neg q_3, \neg q_2,$
$[\beta]\langle\beta^{-1}\rangle q_1 \wedge \langle\gamma^{-1}\rangle(q_3, [\delta](\langle\beta^{-1}\rangle q_1, \langle\gamma^{-1}\rangle(q_2, \langle\delta\rangle(\langle\beta^{-1}\rangle q_1, \langle\beta^{-1}\rangle q_1))))).$

Now,

$A^{\mathbf{p}} = [\alpha](\neg c_1, \neg c_3, \neg c_2,$
$[\beta]\langle\beta^{-1}\rangle c_1 \wedge \langle\gamma^{-1}\rangle(c_3, [\delta](\langle\beta^{-1}\rangle c_1, \langle\gamma^{-1}\rangle(c_2, \langle\delta\rangle(\langle\beta^{-1}\rangle c_1, \langle\beta^{-1}\rangle c_1))))).$

The first-order equivalent of $A^{\mathbf{p}}$, and hence of A, is:

$FO(A^{\mathbf{p}}) = \forall x \forall y_1 \forall y_2 \forall y_3 \forall z (R_\alpha x y_1 y_2 y_3 z \rightarrow \theta_1 \wedge \theta_2)$, where
$\theta_1 = \forall u (R_\beta z u \rightarrow \exists v (R_\beta v u \wedge v = y_1))$ and
$\theta_2 = \exists v_1 \exists v_2 (R_\gamma z v_1 v_2 \wedge v_1 = y_3 \wedge \forall w_1 \forall w_2 (R_\delta v_2 w_1 w_2 \rightarrow \theta_{12} \wedge \theta_{22}))$,
where
$\theta_{12} = \exists t (R_\beta t w_1 \wedge t = y_1)$, and
$\theta_{22} = \exists t_1 \exists t_2 (R_\gamma t_1 w_2 t_2 \wedge t_1 = y_2 \wedge \exists s_1 \exists s_2 (R_\delta t_2 s_1 s_2 \wedge \theta_{221} \wedge \theta_{222}))$,
where $\theta_{221} = \exists t (R_\beta t s_1 \wedge t = y_1)$, and $\theta_{222} = \exists t (R_\beta t s_2 \wedge t = y_1)$.

This formula simplifies to:
$\forall x \forall y_1 \forall y_2 \forall y_3 \forall z (R_\alpha x y_1 y_2 y_3 z \rightarrow \varphi_1 \wedge \exists v (R_\gamma z y_3 v \wedge \forall w_1 \forall w_2 (R_\delta v w_1 w_2 \rightarrow \varphi_2))$,
where
$\varphi_1 = \forall u (R_\beta z u \rightarrow R_\beta y_1 u)$, and
$\varphi_2 = R_\beta y_1 w_1 \wedge \exists t (R_\gamma y_2 w_2 t \wedge \exists s_1 \exists s_2 (R_\delta t s_1 s_2 \wedge R_\beta y_1 s_1 \wedge R_\beta y_1 s_2))$.

As a corollary to Propositions 17 and 25 and Theorem 41 we obtain the main result:

Theorem 43 (Sahlqvist theorem in reversive polyadic hybrid languages)

1. Every PSF in $\mathcal{L}^{\upsilon;\mathbf{n}}_{\tau r}$ or $\mathcal{L}^{@;\mathbf{n}}_{\tau r}$ is first-order definable.

2. Every extension of $\mathcal{K}^{\upsilon;\mathbf{n}}_{\tau r}$ or $\mathcal{K}^{@;\mathbf{n}}_{\tau r}$ with PSFs is complete.

Corollary 44 Every PSF in any polyadic modal language of the types introduced here is first-order definable.

5 Concluding Remarks

5.1 Transfer of Sahlqvist Theorem

We are interested in obtaining a proof of Sahlqvist theorem for hybrid polyadic languages not covered by Theorem 43 by transferring the completeness part of Theorem 43 down to such languages by means of a *proof-theoretic* argument based on conservativeness of the extensions to reversive hybrid languages. There are two main cases to consider: extensions from non-reversive to reversive languages and extensions with nominals. In the case of polyadic languages with υ, but without nominals, Sahlqvist theorem can be proved directly, by adapting the proof for basic purely modal polyadic languages in [10]. However, in arbitrary polyadic languages $\mathcal{L}^{\upsilon;\mathbf{n}}_{\tau}$ or $\mathcal{L}^{D}_{\tau}$ certain complications arise (see §6 in [14] for details) and, although some partial results still hold, the full Sahlqvist theorem for these languages is still open and that justifies the quest for transfer results. In general, neither of the types of extensions mentioned above is always conservative, but here we only consider extensions of logics axiomatized with Sahlqvist formulas. Moreover, in cases when Sahlqvist theorem holds in the weaker languages we *do know that these extensions are conservative.* Besides, reverting the argument, i.e. proving completeness via conservativeness, would have an independent value because it would shed light both on how Sahlqvist formulas work as axioms and how the hybrid mechanisms act over standard modal languages.

5.2 On Kracht Calculus

Kracht has identified in [11] a class of first-order formulas which can be algorithmically translated into Sahlqvist formulas in the classical modal language, and every classical Sahlqvist formula has an equivalent Kracht formula. In a sense, Theorem 43 provides a modal analogue of Kracht calculus: the first-order equivalents of pure formulas in reversive hybrid languages can be easily described syntactically, hence the first-order equivalents of *all* Sahlqvist formulas in these languages can be described and their respective Sahlqvist formulas can be computed effectively by reverting the derivations outlined in the previous section.

5.3 On the Scope of Sahlqvist Theorem

Theorem 43 tells that in reversive hybrid languages all logics axiomatized by Sahlqvist formulas, as defined here, can be axiomatized by pure formulas as well, and for such logics the claims of Sahlqvist theorem are much more transparent. This holds accordingly for all stronger hybrid languages, e.g. those involving *binders* or *quantifiers* over nominals (see [4, 1]): i.e. the pure formulas in those languages subsume, in terms of frame definability and axiomatizability, the respective classes of Sahlqvist formulas.

On the other hand, it is known that this is generally not the case for non-reversive languages. For instance (see [14]), Church-Rosser's formula $\Diamond\Box p \rightarrow \Box\Diamond p$, which is a Sahlqvist formula, is *not equivalent to a pure formula in the basic modal language, because it is not di-persistent in that language.* (A formula is di-persistent if it preserves validity from *discrete general frames* (in which all singletons are definable) to their underlying Kripke frames.) Same formula, however, is di-persistent in a *tense language* and, in fact, it is equivalent to the *pure tense formula* $Fc \rightarrow GFPc$.

Since every pure formula is di-persistent, we have thus re-proved and extended Venema's result about di-persistence of Sahlqvist tense formulas to reversive hybrid polyadic languages.

A fundamental question in this topic is: *what is the largest class of Sahlqvist formulas*? To make this a little more precise, let us emphasize again that all explicit definitions of Sahlqvist formulas given so far are *syntactic* and, in fact, very syntactically sensitive, while the idea behind both parts of Sahlqvist theorem is *semantic*, and it hinges on two basic observations:

- For every Sahlqvist formula A there is a *first-order definable minimal valuation* associated with every Kripke frame F which defines a 'minimal' model M_A^F over F such that $M_A^F \models A$ implies $F \models A$.
- These minimal valuations are *closed* with respect to the topologies associated with descriptive frames, which enforces d-persistence, hence canonicity, of Sahlqvist formulas.

These two properties can be formulated more precisely to give a purely semantic definition of the maximal class of Sahlqvist formulas, simply being the class of those formulas for which the method of proof of Sahlqvist theorem works. However, this definition gives little insight on the syntactic shape of these formulas.

Thus, in conclusion, there is still a gap between the syntactic form and semantic essence of Sahlqvist formulas. Some undecidability results (see [6]) indicate that this gap cannot be closed completely, but still there is justified hope that in the case of reversive hybrid languages syntax and semantics can meet in a large and natural class of formulas as stated in the following conjecture.

Conjecture 45 Every di-persistent and locally first-order definable formula in a reversive language with nominals is axiomatically equivalent to a pure formula, and therefore the classes of Sahlqvist, pure, and locally first-order definable di-persistent formulas in such languages coincide, up to frame equivalence.

Acknowledgments

We thank Carlos Areces and an anonymous referee for useful comments and corrections. The work of Valentin Goranko was partly done during his visit to the University of Sofia and was supported by the SASOL research fund of the Faculty of Natural Sciences of Rand Afrikaans University and by a research grant GUN 2034353 of the National Research Foundation of South Africa.

References

[1] C. Areces. *Logic Engineering. The case of Description and Hybrid Logics, ILLC Dissertation Series DS-2000-5.* PhD thesis, ILLC, University of Amsterdam, 2000.

[2] P. Blackburn. Internalizing labeled deduction. *Journal of Logic and Computation*, 10(1):137–168, 2000.

[3] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic.* Number 53 in Cabridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[4] P. Blackburn and J. Seligman. What are hybrid languages? In M. Kracht, M. de Rijke, H. Wansing, and M. Zakcharyaschev, editors, *Advances in Modal Logic, vol. 1*, pages 41–62, Stanford, 1998. CSLI Publications.

[5] P. Blackburn and M. Tzakova. Hybrid completeness. *Logic Journal of the IGPL*, 4:625–650, 1998.

[6] A. Chagrov and M. Zakharyaschev. Sahlqvist formulas are not so elementary even above S4. In L. Csirmaz, D. Gabbay, and M. de Rijke, editors, *Logic Colloquium'92*, pages 61–73, Stanford, 1995. CSLI Publications.

[7] M. de Rijke. *Extending Modal Logic.* PhD thesis, ILLC, University of Amsterdam, 1993.

[8] G. Gargov and V. Goranko. Modal logics with names. *Journal of Philosophical Logic*, 22(6):607–636, 1993.

[9] V. Goranko and S. Passy. Using the universal modality: Gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992.

[10] V. Goranko and D. Vakarelov. Sahlqvist formulas unleashed in polyadic modal languages. In F. Wolter, H. Wansing, M. de Rijke, and M. Zakcharyaschev, editors, *Advances in Modal Logic, vol. 3*, Stanford, 2001. to appear in: CSLI Publications.

[11] M. Kracht. How completeness and correspondence theory got married. In M. de Rijke, editor, *Diamonds and Defaults*, pages 175–214. Kluwer, Dordrecht, 1993.

[12] S. Passy and T. Tinchev. An essay in combinatory dynamic logic. *Information and Computation*, 93:263–332, 1991.

[13] D. Vakarelov. Rough polyadic modal logics. *Journal of Applied Non-classical Logics*, 1:9–35, 1991.

[14] Y. Venema. Derivation rules as anti-axioms in modal logic. *Journal of Symbolic Logic*, 58:1003–1034, 1993.