THE UNIVERSITY
of MANCHESTER

January 2003

Computer Science
University of Manchester

Mechanised Reasoning and Model Generation for Extended Modal Logics

Renate A. Schmidt and Ullrich Hustadt

Department of Computer Science
University of Manchester
Preprint Series
CSPP-19

Mechanised Reasoning and Model Generation for Extended Modal Logics

Renate A. Schmidt

Department of Computer Science, University of Manchester, United Kingdom
schmidt@cs.man.ac.uk

Ullrich Hustadt

Department of Computer Science, University of Liverpool, United Kingdom
U.Hustadt@csc.liv.ac.uk

January 2003

Abstract

The approach presented in this overview paper exploits that modal logics can be seen to be fragments of first-order logic and deductive methods can be developed and studied within the framework of first-order resolution. We focus on a class of extended modal logics very similar in spirit to propositional dynamic logic and closely related to description logics. We review and discuss the development of decision procedures for decidable extended modal logics and look at methods for automatically generating models.

Recent preprints issued by the Department of Computer Science, Manchester University, are available on WWW via URL `http://www.cs.man.ac.uk/preprints/index.html` or by ftp from `ftp.cs.man.ac.uk` in the directory `pub/preprints`.

1 Introduction

Over the last nearly ten years a variety of methods for reasoning with modal and description logics have been developed, implemented and applied in several case studies, cf. for example [31, 36, 41, 42, 37]. Though the logics involved are very similar, the reasoning methods used and proof search strategies employed can differ considerably. Various empiricical studies have been undertaken mainly for basic multi-modal logic or its corresponding description logic $\mathcal{ALC}$. Many of these studies are competitive in nature or study the affects of various optimisation methodologies on the performance of provers. While such work is vital, it is only a beginning. From current studies it is difficult to extrapolate general conclusions for different reasoning methods and different logics not considered in such studies. The literature on formal logic and proof theory does not give much guidance either. Usually particular forms of reasoning methods tend to get favoured over others, mostly due to ease of presentation, without there being theoretical or empirical evidence for their practical usefulness. Thus currently it is still difficult to choose between the different methods and provers, for there is no general body of knowledge which would support well-judged choices. In the area of automated reasoning for propositional logics there have been extensive analytical and empirical evaluations. Similar research developments for modal and description logic reasoning systems are only starting to get off the ground.

The aim of this paper is to give a overview of some recent advances. We concentrate on decision procedures developed in the framework of first-order resolution and focus on translation-based resolution methods for modal logics. This means that we take a modal formula, translate it into first-order logic through the Kripke-semantics, and then apply some variant of resolution to it. Some obvious advantages of using the combination of a translation method and resolution are the following. Any modal logic which can be embedded into first-order logic can be treated. The translations are straightforward, and can be performed in time $O(n \log n)$, so no engineering effort is needed here. For the resolution part, standard resolution provers can be used, or otherwise they can be used with small adaptations. Modern resolution provers are among the most sophisticated and fastest theorem provers available. The translation approach is generic, it can handle first-order modal logics, undecidable (first-order definable) modal logics, and combinations of modal and non-modal logics. In all cases soundness and completeness of the approach is immediate from the soundness and completeness of the translation mapping and the resolution calculus. Resolution provers provide decision procedures for a large class of (extended) modal logics and description logics. Often the same refinements that decide modal and description logics decide also more expressive first-order generalisations such as the guarded fragment or Maslov's class K [20, 43].

This survey focusses on the extended modal logic $K_{(m)}(\cap, \cup, ^{-}, \smile)$, first considered in De Nivelle, Hustadt and Schmidt [13], and subsystems thereof as well as extensions with relational theories. $K_{(m)}(\cap, \cup, ^{-}, \smile)$ is a PDL-like logic which permits complex formulae as parameters of the modal operators. This is useful for application domains in artificial intelligence and computational linguistics. For example, if e denotes the eats relation and p is the set of plants, then $\langle e \rangle p$ can be interpreted as denoting the set of plant eaters, while $[e]p$ denotes the set of vegetarians, who eat nothing but plant matter.[1] An expression which requires complex relational parameters is the set of cheese lovers, given by $\langle e \wedge l \rangle c \wedge \neg\langle \neg(e \wedge c) \rangle c$, where l denotes the 'likes' relation and c is interpreted as the set of cheeses.[2]

Formally, $K_{(m)}(\cap, \cup, ^-, ^\smile)$ is the multi-modal logic defined over families of relations closed under intersection, union, complementation and converse. It extends Boolean modal logic (due to Gargov and Passy [24]) with converse on relations. $K_{(m)}(\cap, \cup, ^-, ^\smile)$ is very expressive. It subsumes standard modal logics such as *K*, *KT*, *KD*, *KB*, *KTB*, and *KDB*, their independent joins, as well as the basic tense logic K_t. Global satisfiability of these logics can be embedded in $K_{(m)}(\cap, \cup, ^-, ^\smile)$ and it subsumes modal logics extended with the universal modality. Logics of philosophical interest such as logics expressing inaccessibility, sufficiency, or both necessity and sufficiency [25, 38, 39] can be embedded in $K_{(m)}(\cap, \cup, ^-, ^\smile)$. Certain forms of interactions and correspondence properties, for example, inclusions among relations and symmetry, are covered as well. $K_{(m)}(\cap, \cup, ^-, ^\smile)$ subsumes a large class of well-known description logics [13, 29]. It is most closely related to the description logic $\mathcal{ALB}$ introduced in [45].

This paper gives an overview of different resolution refinements which provide decision procedures for first-order fragments corresponding to a variety of extended modal logics. We focus on fragments induced by the standard relational translation of modal logics. Other translation methods exist but, as yet, it is not known how to treat modal logics with complex modal parameters within the context of these translation methods.

Regardless as to which translation method is adopted, a crucial decision is the choice of a suitable refinement of the basic resolution calculus for first-order logic. Depending on our aims we have various options. Ordering refinements provide decision procedures for very expressive logics, while if we are interested in generating models for satisfiable formulae selection-based refinements (or hyperresolution) are more natural (Fermüller et al. [18, 17], Leitsch [50], Hustadt et al [30, 45, 44, 47]). We discuss an ordered resolution decision procedure for a class of clauses induced by $K_{(m)}(\cap, \cup, ^-, ^\smile)$ in Section 6. In Section 7 we describe a refinement which relies solely on the selection of negative literals for certain extensions of $K_{(m)}(\cap, \cup, ^\smile)$. This refinement has the property that for many modal logics its derivations resemble those of tableau calculi. We consider the polynomial simulation of single-step prefix tableau by selection-based resolution in Section 8. Such simulation results do not only say something about the relative complexity of resolution and tableaux, they can also be exploited to transfer deduction methodologies and optimisation techniques between the different approaches. Moreover, the relationship can be exploited for extracting new tableau calculi from resolution in a more or less automatic way. As a case analysis, in Section 9, we define a semantic tableau calculus for the logic $K_{(m)}(\cap, \cup, ^\smile)$ which is derived from the selection-based resolution procedure. Soundness,

[1] $x \models [e]p$ iff for any y, such that $R_e(x, y)$ we have that $y \models p$. Thus, the meaning of $x \models [e]p$ is 'everything that x eats is plant matter'.

[2] Observe $x \models \langle e \wedge l \rangle c$ iff x eats and likes (some) cheese. Further, $x \models \neg\langle \neg(e \wedge c) \rangle c$ iff for any $y \models c$, both $R_e(x, y)$ and $R_l(x, y)$ are true. Therefore, cheese lovers are people who eat and like every cheese.

completeness and termination results are then mear corollaries of corresponding results for the resolution refinement. The selection-based refinement also has the property that, like tableau-based procedures, it can be used for the automatic construction of models for satisfiable formulae and the models are finite if the procedure is a decision procedure. This is the topic of Section 10. In Section 11 we mention automated reasoning tools that implement the procedures described in this paper.

Before we can proceed to the main part of this paper, namely Sections 6–11, we need to define the class of logics under consideration, how they translate to first-order logic and describe the resolution framework. This is done in Sections 3– 5. Section 2 summarises the notational conventions used in this paper. The final section mentions some important topics not covered in this paper (due to space restrictions).

2 Notational convention

Throughout the notational convention is the following. The letters x, y, z are reserved for first-order variables, s, t, u, v for terms, a, b for constants, f, g, h for function symbols, and p, q, r for propositional symbols, and P, Q, R for predicate symbols. A is the letter reserved for atoms, L for literals, and C, D for clauses. For sets of clauses the letter N is used. The Greek letters φ, ψ, ϕ are reserved for modal or first-order formulae, and α, β, γ are reserved for relational formulae.

3 Extended modal logics

The language of $K_{(m)}(\cap, \cup, ^-, ^\smile)$ is defined over countably many propositional variables $p, p_1, p_2, \ldots$, and countably many relational variables $r, r_1, r_2, \ldots$. A *propositional atom* is a propositional variable, $\top$ or $\bot$. A *modal formula* is either a propositional atom or a formula of the form $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\langle\alpha\rangle\varphi$ and $[\alpha]\varphi$, where φ is a modal formula and α is a relational formula. A *relational formula* is a relational variable or has one of the following forms: $\alpha \wedge \beta$, $\alpha \vee \beta$, $\neg\alpha$, and $\alpha^\smile$ (converse), where α and β are relational formulae. Other connectives are defined to be abbreviations, for example, $\varphi \to \psi = \neg\varphi \vee \psi$ or the universal modality is $[u] = [r_j \vee \neg r_j]$, for some relational variable r_j.

The semantics of $K_{(m)}(\cap, \cup, ^-, ^\smile)$ is defined in terms of relational structures or frames. A frame is a tuple (W, R) of a non-empty set W (of worlds) and a mapping R from relational formulae to binary relations over W satisfying:

$$R_{\alpha\wedge\beta} = R_\alpha \cap R_\beta \qquad R_{\alpha\vee\beta} = R_\alpha \cup R_\beta \qquad R_{\neg\alpha} = \overline{R_\alpha} \qquad R_{\alpha^\smile} = R_\alpha^\smile.$$

Here and in the rest of the paper we prefer to use the notation R_α instead of $R(\alpha)$. The defining class of frames of a modal logic determines, and is determined by, a corresponding class of models. A model (an interpretation) is given by a triple $\mathcal{M} = (W, R, \iota)$, where

Seriality, for $D = \langle r \rangle \top$:	$\forall x \exists y \, R(x,y)$
Reflexivity, for $T = [r]p \rightarrow p$:	$\forall x \, R(x,x)$
Symmetry, for $B = \langle r \rangle [r]p \rightarrow p$:	$\forall x \forall y \, (R(x,y) \rightarrow R(y,x))$
Inclusion, for $[r_1]p \rightarrow [r_2]p$:	$\forall x \forall y \, (R_2(x,y) \rightarrow R_1(x,y))$
For $[r_3]p \rightarrow ([r_1]p \vee [r_2^{\smile}]p)$:	$\forall x \forall y \, (R_1(x,y) \wedge R_2(y,x) \rightarrow R_3(x,y))$

Figure 1: Some correspondence properties

(W, R) is a frame and ι is a mapping from modal formulae to subsets of W satisfying:

$$\begin{array}{lll} \iota(\bot) = \emptyset & \iota(\top) = W & \iota(\neg\varphi) = \overline{\iota(\varphi)} \\ \iota(\varphi \wedge \psi) = \iota(\varphi) \cap \iota(\psi) & \multicolumn{2}{l}{\iota(\langle\alpha\rangle\varphi) = \{x \mid \exists y \in W \, ((x,y) \in R_\alpha \wedge y \in \iota(\varphi))\}} \\ \iota(\varphi \vee \psi) = \iota(\varphi) \cup \iota(\psi) & \multicolumn{2}{l}{\iota([\alpha]\varphi) = \{x \mid \forall y \in W \, ((x,y) \in R_\alpha \rightarrow y \in \iota(\varphi))\}.} \end{array}$$

A modal formula is satisfiable iff an $\mathcal{M}$ exists such that for some x in W, $x \in \iota(\varphi)$.

We also consider logics with fewer relational operations, as well as logics restricted by relational theories consisting of additional frame properties. A logic $K_{(m)}(\star_1, \ldots, \star_k)$ *in-between* K *and* $K_{(m)}(\cap, \cup, ^-, ^\smile)$, where the $\star_i$ are distinct operations from $\{\cap, \cup, ^-, ^\smile\}$ ($m \geq 1$, $0 \leq i \leq k \leq 4$), is defined to be the multi-modal logic defined over relations closed under $\star_1, \ldots, \star_k$. If L is a logic in-between K and $K_{(m)}(\cap, \cup, ^-, ^\smile)$ and Δ is a set of relational frame properties then $L\Delta$ denotes the logic characterised by the class of L-frames which satisfy the conjunction of properties in Δ.[3] Examples of relational frame properties and the corresponding modal axiom schemas are given in Figure 1.

It is well-known that an implication between relational formulae can be defined by $(\alpha \rightarrow \beta) = [\alpha \wedge \neg\beta]\bot$ in Boolean modal logic [64] and therefore also $K_{(m)}(\cap, \cup, ^-, ^\smile)$. For example in $K_{(m)}(\cap, \cup, ^-, ^\smile)$ the symmetry of the accessibility relation R_1 associated with r_1 can be specified by $r_1 \rightarrow r_1^\smile$. We also observe that a modal logic L including relational complementation and one relational conjunction and relational disjunction allows for the definition of the universal modality. If r is a relational name in the language of the logic L then the universal modality $[u]$ can be defined by $[u]\varphi = [r \vee \neg r]\varphi = [\neg(r \wedge \neg r)]\varphi$.

4 Translation to first-order logic

Modal formulae will be mapped to first-order logic formulae by two transformations: a translation of the modal formula into first-order logic, in this case, a semantics-based translation, followed by a structural transformation.

The *standard semantics-based translation* of $K_{(m)}(\cap, \cup, ^-, ^\smile)$ into first-order logic is determined by the definition of the semantics of the logical operators. For modal formulae

[3] Used in a formula Δ is assumed to represent the conjunction of relational properties.

the translation is specified by the following.

$$\begin{array}{rlrl} \pi(\top, x) &= \top & \pi(\bot, x) &= \bot \\ \pi(p_i, x) &= P_i(x) & \pi(\neg\varphi, x) &= \neg\pi(\varphi, x) \\ \pi(\varphi \star \psi, x) &= \pi(\varphi, x) \star \pi(\psi, x) & \text{for } \star \in \{\wedge, \vee, \rightarrow, \leftrightarrow\} & \\ \pi(\langle\alpha\rangle\varphi, x) &= \exists y\,(\tau(\alpha, x, y) \wedge \pi(\varphi, y)) & \pi([\alpha]\varphi, x) &= \forall y\,(\tau(\alpha, x, y) \rightarrow \pi(\varphi, y)) \end{array}$$

Relational formulae are translated according to the following.

$$\begin{array}{lll} \tau(r_j, x, y) = R_j(x, y) & \tau(\neg\alpha, x, y) = \neg\tau(\alpha, x, y) & \tau(\alpha^{\smile}, x, y) = \tau(\alpha, y, x) \\ \tau(\alpha \star \beta, x, y) = \tau(\alpha, x, y) \star \tau(\beta, x, y) & \text{for } \star \in \{\wedge, \vee\} & \end{array}$$

In the translation each propositional or relational variable (p_i or r_j) is uniquely associated with a unary or binary predicate variable, denoted by the corresponding capital letter (P_i or R_j).

By definition, Π maps any modal formula φ to $\exists x\ \pi(\varphi, x)$.

Theorem 4.1 *Let L be a logic in-between K and $K_{(m)}(\cap, \cup, ^-, ^\smile)$ and Δ a (possibly empty) set of relational frame properties. For any modal formula φ, φ is satisfiable in $L\Delta$ iff $\Delta \wedge \Pi(\varphi)$ is first-order satisfiable.*

The purpose of the structural transformation is to convert the first-order translation into a more managable form. Before we describe it formally, we need to state some definitions of basic notions.

The polarity of (occurrences of) modal or first-order subformulae is defined as usual. Any occurrence of a proper subformula of an equivalence has *zero polarity.* For occurrences of subformulae not below a '$\leftrightarrow$' symbol, an occurrence of a subformula has *positive polarity* if it is one inside the scope of an even number of (explicit or implicit) negations, and it has *negative polarity* if it is one inside the scope of an odd number of negations.

For any first-order formula φ, if λ is the position of a subformula in φ, then $\varphi|_\lambda$ denotes the subformula of φ at position λ and $\varphi[\psi \mapsto \lambda]$ is the result of replacing $\varphi|_\lambda$ at position λ by ψ. The set of all the positions of subformulae of φ is denoted by $\mathrm{Pos}(\varphi)$.

Structural transformation, also referred to as *renaming*, associates with each element λ of $\Lambda \subseteq \mathrm{Pos}(\varphi)$ a predicate symbol Q_λ and a literal $Q_\lambda(\overline{x})$, where $\overline{x} = x_1, \ldots, x_n$ are the free variables of $\varphi|_\lambda$, the symbol Q_λ does not occur in φ and two symbols Q_λ and $Q_{\lambda'}$ are equal only if $\varphi|_\lambda$ and $\varphi|_{\lambda'}$ are equivalent formulae.[4] Let $\mathrm{Def}^+_\lambda(\varphi) = \forall\overline{x}\,(Q_\lambda(\overline{x}) \rightarrow \varphi|_\lambda)$ and $\mathrm{Def}^-_\lambda(\varphi) = \forall\overline{x}\,(\varphi|_\lambda \rightarrow Q_\lambda(\overline{x}))$. The *definition* of Q_λ is the formula

$$\mathrm{Def}_\lambda(\varphi) = \begin{cases} \mathrm{Def}^+_\lambda(\varphi) & \text{if } \varphi|_\lambda \text{ has positive polarity,} \\ \mathrm{Def}^-_\lambda(\varphi) & \text{if } \varphi|_\lambda \text{ has negative polarity,} \\ \mathrm{Def}^+_\lambda(\varphi) \wedge \mathrm{Def}^-_\lambda(\varphi) & \text{otherwise.} \end{cases}$$

[4]In practice, one may want to use the same symbols for variant subformulae, or subformulae which are obviously equivalent, for example, $\varphi \vee \varphi$ and φ.

The corresponding clauses are called *definitional clauses.* Now, define $\mathrm{Def}_\Lambda(\varphi)$ inductively by: $\mathrm{Def}_\emptyset(\varphi) = \varphi$ and

$$\mathrm{Def}_{\Lambda\cup\{\lambda\}}(\varphi) = \mathrm{Def}_\Lambda(\varphi[Q_\lambda(\overline{x}) \mapsto \lambda]) \wedge \mathrm{Def}_\lambda(\varphi),$$

where λ is maximal in $\Lambda \cup \{\lambda\}$ with respect to the prefix ordering on positions. A *definitional form* of φ is $\mathrm{Def}_\Lambda(\varphi)$, where Λ is a subset of all positions of subformulae (usually, non-atomic or non-literal subformulae).

Theorem 4.2 (e.g. [7, 65]) *Let φ be a first-order formula. (i) φ is satisfiable iff $\mathrm{Def}_\Lambda(\varphi)$ is satisfiable, for any $\Lambda \subseteq \mathrm{Pos}(\varphi)$. (ii) $\mathrm{Def}_\Lambda(\varphi)$ can be computed in linear time.*

5 First-order resolution

Basics. The usual definition of clausal logic is assumed. A *literal* is an atom or the negation of an atom. The former is said to be a *positive literal* and the latter a *negative literal.* If the predicate symbol of a literal has arity one (two) then we call this literal a *unary literal* (*binary literal*). A clause with one literal is a *unit clause* (or unit). If this literal is a unary (binary) literal then the clause will be called a unary (binary) unit clause. In this paper *clauses* are assumed to be multisets of literals, and will be denoted by $P(x) \vee P(x) \vee \neg R(x, y)$, for example. The empty clause will be denoted by $\emptyset$. The components in the variable partition of a clause are called *variable-disjoint* or *split components*, that is, split components do not share variables. A clause which cannot be split further will be called a *maximally split clause.* A *positive* (resp. *negative*) clause contains only *positive* (resp. *negative*) literals.

Two formulae or clauses are said to be *variants* of each other if they are equal modulo variable renaming. Variant clauses are assumed to be equal.

We say an expression is *functional* if it contains a constant or a non-nullary function symbol. Otherwise it is called *non-functional.*

Resolution. Now, we briefly recall the definition of ordered resolution extended with a selection function from Bachmair et al [2, 3, 4]. Derivations are controlled by an admissible ordering $\succ$ and a selection function. Basically the idea is that inferences are restricted to literals maximal under the ordering $\succ$ while the selection function is used to override the ordering, and give preference to inferences with negative literals. A third parameter in our presentation is a normalisation function Norm.

By definition, an ordering $\succ$ is *admissible*, if (i) it is a total, well-founded ordering on the set of ground literals, (ii) for any atoms A and B, it satisfies: $\neg A \succ A$, and $B \succ A$ implies $B \succ \neg A$, and (iii) it is stable under the application of substitutions. An ordering is said to be *liftable* if it satisfies (iii). The multiset extension of $\succ$ provides an admissible ordering on clauses. A literal L is said to be *(strictly) maximal* with respect to a clause C if for any literal L' in C, $L' \not\succ L$ ($L' \not\succeq L$). Let M be a set and $\succ_c$ an arbitrary ordering on M. Assume that with every literal L we associate a complexity measure $c_L \in M$. An

Deduce: $$\frac{N}{N \cup \{\textsc{Norm}(C)\}}$$ if C is a factor or resolvent of premises in N.

Delete: $$\frac{N \cup \{C\}}{N}$$ if C is redundant.

Split: $$\frac{N \cup \{C \vee D\}}{N \cup \{C\} \mid N \cup \{D\}}$$ if C and D are variable-disjoint.

Resolvents and factors are computed with:

Ordered resolution: $$\frac{C \vee A_1 \quad \neg A_2 \vee D}{(C \vee D)\sigma}$$

provided (i) σ is the most general unifier of A_1 and A_2, (ii) no literal is selected in C, and $A_1\sigma$ is strictly $\succ$-maximal with respect to $C\sigma$, and (iii) $\neg A_2$ is either selected, or $\neg A_2\sigma$ is maximal with respect to $D\sigma$ and no literal is selected in D.

Ordered factoring: $$\frac{C \vee A_1 \vee A_2}{(C \vee A_1)\sigma}$$

provided (i) σ is the most general unifier of A_1 and A_2, and (ii) no literal is selected in C and $A_1\sigma$ is $\succ$-maximal with respect to $C\sigma$.

Figure 2: The calculus R

ordering is *compatible with a given complexity measure* c_L on ground literals, if $c_L \succ_c c_{L'}$ implies $L \succ L'$ for any two ground literals L and L'.

A *selection* function S assigns to each clause a possibly empty set of occurrences of negative literals. If C is a clause, then the literal occurrences in $S(C)$ are *selected*. No restrictions are imposed on the selection function. The minimal requirement for the normalisation function is that $\textsc{Norm}(C)$ is a clause which is logically equivalent to C and $\textsc{Norm}(C) \preceq C$. Many resolution decision procedures rely on condensing (defined below) as the minimal normalisation function.

Let R be the resolution calculus defined by the rules of Figure 2. As is usual we implicitly assume that the premises of the resolution rule have no common variables. The premise $C \vee A_1$ of the resolution rule and premise of the factoring rule will be referred to as a *positive premise*, while the premise $\neg A_2 \vee D$ of the resolution rule will be referred to as a *negative premise*. The literals resolved upon and factored upon are called *eligible literals*.

The *splitting rule* is a rule familiar from DPLL algorithms and tableau calculi. Instead of trying to refute $N \cup \{C \vee D\}$ one tries to refute $N \cup \{C\}$ and $N \cup \{D\}$ (or $N \cup \{C\}$ and $N \cup \{D, \neg C\}$, if C is a ground clause). The splitting rule is don't know non-deterministic and requires backtracking. However, in the resolution context splitting can be simulated by introducing a new propositional symbol. If $C \vee D$ is a clause that can be split into two split components C and D, then it is possible to replace $C \vee D$ by two clauses $C \vee q$, and $\neg q \vee D$. q is made minimal in the ordering, and $\neg q$ is selected [11, 67]. In most cases this is easier to implement than the full splitting rule.

R forms a complete refutation system for clause sets. In general, the calculus R can be enhanced with standard simplification rules such as tautology deletion and subsumption deletion, in fact, it can be enhanced by any simplification rule which is compatible with a general notion of redundancy [3, 4]. Essentially, a ground clause is redundant in a set N with respect to the ordering $\succ$ if it follows from smaller instances of clauses in N, and a non-ground clause is redundant in N if all its ground instances are redundant in N. A set N of clauses is *saturated up to redundancy* with respect to a particular refinement of resolution if the conclusion of every inference from non-redundant premises in N is either contained in N, or else is redundant in N. Subsumption and condensing are instances of redundancy elimination. A clause D *subsumes* a clause C iff there exists a substitution σ such that $D\sigma \subseteq C$ (strictly speaking, in our framework $D\sigma \subset C$ has to hold). The *condensation* $\textsc{Cond}(C)$ of a clause C is a minimal[5] multiple factor of C which subsumes C. A clause C is *condensed* if there is no proper subclause of C which is a factor of C.

A *derivation* in R from a set of clauses N is a finitely branching, ordered tree T with root N and nodes which are sets of clauses. The tree is constructed by applications of the expansion rules to the leaves. We assume that no resolution or factoring inference is computed twice on the same branch of the derivation. Any path $N(= N_0), N_1, \ldots$ in a derivation T is called a *closed branch* in T iff the clause set $\bigcup_{j\geq 0} N_j$ contains the empty clause, otherwise it is called an *open branch*. We call a branch B in a derivation tree *complete* (with respect to R) iff no new successor nodes can be added to the endpoint of B by R, otherwise it is called an *incomplete branch*. A derivation T is a *refutation* iff every path $N(= N_0), N_1, \ldots$ in it is a closed branch, otherwise it is called an *open derivation*.

A derivation T from N is called *fair* iff for any path $N(= N_0), N_1, \ldots$ in T, with *limit* $N_\infty = \bigcup_{j\geq 0}\bigcap_{k\geq j} N_k$, it is the case that each clause C which can be deduced from non-redundant premises in N_∞ is contained in some N_j. Intuitively, fairness means that no non-redundant inferences are delayed indefinitely. For a finite complete branch $N(= N_0), N_1, \ldots N_n$, the limit N_∞ is equal to N_n.

Theorem 5.1 ([4]) *Let T be a fair R derivation from a set N of clauses. Then: (i) If $N(= N_0), N_1, \ldots$ is a path with limit N_∞, then N_∞ is saturated (up to redundancy). (ii) N is satisfiable if and only if there exists a path in T with limit N_∞ such that N_∞ is satisfiable. (iii) N is unsatisfiable if and only if for every path $N(= N_0), N_1, \ldots$ the clause set $\bigcup_{j\geq 0} N_j$ contains the empty clause.*

It should be noted that inferences with ineligible literals are not forbidden, but are provably redundant. In other words, only inferences with eligible literals need to be performed for soundness and completeness.

[5] Minimality is with respect to the number of literals in the clause.

6 Decision procedures using ordered resolution

Many modal logics naturally translate into decidable fragments of first-order logic. For example the basic logic K translates into the two-variable fragment, into the guarded fragment, into Maslov's class K, and into fluted logic [29]. By constructing decision procedures for these decidable fragments, one obtains generic decision procedures for modal logics and the corresponding description logics. Resolution decision procedures have been developed for the guarded fragment [12, 20], for Maslov's class K [43], for fluted logic [71] and various other classes related to modal logics, see e.g. [17, 29, 40]. In this paper we consider only the relationship to a fragment of clausal logic based on the two-variable fragment. The fragment is called DL^* [13]. It is a variation of the class of DL-clauses, that was introduced in [45] with the purpose of handling expressive description logics.

In order to simplify the definition of the fragment DL^* of clausal logic all clauses are assumed to be maximally split. The notions can be easily adopted for clauses with more than one split component. A maximally split clause C is a *DL^*-clause* if the following conditions are satisfied.

1. All literals are unary, or binary.

2. There is no nesting of function symbols.

3. Every functional term in C contains all the variables of C. (This condition implies that if C contains a functional ground term, then C is ground.)

4. Every binary literal (even if it has no functional terms) contains all the variables of C.

Examples of DL^*-clauses include ground clauses, and the following.

$$\begin{array}{ll}
\neg Q_0(x) \vee Q_1(x) \vee \neg Q_2(x) & Q_0(x) \vee \neg R_0(x,y) \vee Q_1(y) \\
\neg Q_0(x) \vee Q_1(f(x)) & R_0(x,y) \vee \neg R_1(y,x) \vee R_2(x,y) \\
\neg Q_0(x) \vee \neg R_0(f(x),x) & R_0(x,y) \vee \neg R_1(x,f(x,y)) \vee R_2(f(x,y),y)
\end{array}$$

The clauses $R_0(x,y) \vee R_0(x,f(x))$, $Q_0(x,x,x) \vee Q_1(f(f(x)))$ and $R_0(x,x) \vee R_1(x,y)$ do not belong to the class of DL^*-clauses. The clause $Q_0(x) \vee Q_1(a)$ does not belong to DL^*, since it is not maximally split.

Theorem 6.1 ([13, 45]) *Over a finite signature*[6] *there are only finitely many maximally split DL^*-clauses (modulo variable renaming).*

The proof can be obtained by first observing that there is a fixed upper bound for the maximal number of variables in a clause. Then there are only a finite number of possible literals. Because every clause is a subset of the set of possible literals, there is a finite set of possible clauses.

[6] The supply of function symbols and predicate symbols is finite, while there are possibly infinite but countably many variables.

The first-order translation of the modal formula $[\neg r_1 \wedge r_2]\langle \neg r_1 \wedge r_2 \rangle p$ is

$$\exists x \forall y\, ((\neg R_1(x,y) \wedge R_2(x,y)) \to \exists z\, (\neg R_1(y,z) \wedge R_2(y,z) \wedge P(z))).$$

The structural transformation results in the set of formulae on the left, while the clausal form is given on the right. (Here α is used as an abbreviation for $\neg r_1 \wedge r_2$.)

$\exists x\, Q_{[\alpha]\langle\alpha\rangle p}(x)$	$Q_{[\alpha]\langle\alpha\rangle p}(a)^*$
$\forall x\, (Q_{[\alpha]\langle\alpha\rangle p}(x) \to$	$\neg Q_{[\alpha]\langle\alpha\rangle p}(x) \vee \neg Q_\alpha(x,y)^* \vee Q_{\langle\alpha\rangle p}(y)$
$\quad \forall y\, (Q_\alpha(x,y) \to Q_{\langle\alpha\rangle p}(y)))$	$\neg Q_{\langle\alpha\rangle p}(x) \vee Q_\alpha(x, f(x))^*$
$\forall x\, (Q_{\langle\alpha\rangle p}(x) \to \exists y\, (Q_\alpha(x,y) \wedge P(y)))$	$\neg Q_{\langle\alpha\rangle p}(x) \vee P(f(x))^*$
$\forall xy\, (Q_\alpha(x,y) \to (\neg R_1(x,y) \wedge R_2(x,y)))$	$\neg Q_\alpha(x,y)^* \vee \neg R_1(x,y)^*$
$\forall xy\, ((\neg R_1(x,y) \wedge R_2(x,y)) \to Q_\alpha(x,y))$	$\neg Q_\alpha(x,y)^* \vee R_2(x,y)^*$
	$R_1(x,y)^* \vee \neg R_2(x,y)^* \vee Q_\alpha(x,y)^*$

Figure 3: A sample transformation of a modal logic formula to DL^*

Theorem 6.2 ([13]) *The number of possible DL^*-clauses is bounded by $2^{2^{f(s)}}$, where f is of order $s\log(s)$ and s is the size of the signature.*

The reduction of modal formulae to sets of DL^*-clauses makes use of a structural transformation introducing new names for subformulae corresponding to non-atomic subexpressions of the original modal formula [13, 45]. The reduction is illustrated in Figure 3. It is not difficult to verify that the generated clauses are all DL^* clauses. In general, it can be proved that:

Theorem 6.3 ([13]) *Let φ' be a first-order formula that results from translation of a modal formula φ in $K_{(m)}(\cap, \cup, ^-, ^\smile)$. Every clause in the clausal normal form of $\mathrm{Def}_\Lambda(\varphi')$ is a DL^*-clause, where $\Lambda = \{\lambda \mid$ there is a non-atomic subexpression $\varphi|_{\lambda'}$ of φ and $\varphi'|_\lambda = \Pi(\varphi|_{\lambda'})\}$.*

In order to decide the class DL^*, we use the following ordering which is similar to the recursive path ordering. First we define an order $>_d$ on terms: $s >_d t$ if s is deeper than t, and every variable that occurs in t, occurs deeper in s. Then we define $P(s_1, \ldots, s_n) \succ Q(t_1, \ldots, t_m)$ as $\{s_1, \ldots, s_n\} >_d^{\mathrm{mul}} \{t_1, \ldots, t_m\}$. Here $>_d^{\mathrm{mul}}$ is the multiset extension of $>_d$. So we have $P(f(x)) \succ P(a), P(x)$ and $P(x,y) \succ Q(x)$, but not $P(f(x)) \succ P(f(a))$. The ordering $>_d$ originates from Fermüller et al. [18]. The selection function S does not select any negative literal in any clause. We denote this particular instance of the resolution calculus R by $\mathsf{R}^{\mathsf{ord}}$.

In the example in Figure 3 the maximal literals are marked with *. These are the literals that can potentially be resolved or factored upon.

In order to prove that the procedure $\mathsf{R}^{\mathsf{ord}}$ is indeed a decision procedure we have to show that it is complete, and terminating. The completeness follows from Theorem 5.1.

Termination is a consequence of Theorem 6.1, and the fact that the restriction derives only clauses that are within DL^*, or splittable clauses with split components in DL^* (cf. [40, 45]).

Theorem 6.4 *Let L be a logic in-between K and $K_{(m)}(\cap, \cup, ^-, ^\smile)$. Let Δ be a finite set of relational properties expressible in DL^*. Let N be the clausal form of $\Delta \wedge \mathrm{Def}_\Lambda \Pi(\varphi)$, where φ is any modal formula in L and Λ is defined as in Theorem 6.3. Then: (i) Any derivation from N in R^{ord} (up to redundancy) terminates in double exponential time. (ii) φ is unsatisfiable in L iff there is a refutation of N in R^{ord}.*

Relational properties expressible in DL^* include the Boolean combination of relational inclusions or equivalences expressed over intersection, union, complementation and converse. Moreover, reflexivity and irreflexivity can be expressed in DL^*. It is usually the case that when studying modal decidability problems by analysing the decidability of related clausal classes one comes to realise that stronger results are possible than initially anticipated. For instance, it is not difficult to see that modal and relational formulae with positive occurrences of relational composition can also be embedded into the class DL^*. This means Theorem 6.4 can be strengthened. Let $K_{(m)}(\cap, \cup, ^-, ^\smile, \upharpoonleft, ;^{\mathsf{pos}})$ denote the multi-modal logic in which relational formulae may also have the forms: $\alpha \upharpoonleft \varphi$ (domain restriction), and $\alpha \,;\beta$ (composition), but the latter may occur positively only (that is, occur the scope of an even number of explicit and implicit negation symbols). The semantics of the new operators are defined (as expected) by: $R_{\alpha \upharpoonleft \varphi} = \{(x, y) \mid (x, y) \in R_\alpha \wedge x \in \iota(\varphi)\}$, and $R_{\alpha\,;\beta} = R_\alpha \,; R_\beta = \{(x, y) \mid \exists z\,((x, z) \in R_\alpha \wedge (z, y) \in R_\beta)\}$. Observe that the range restriction of a relation can be represented in terms of domain restriction and converse, by $(\alpha^\smile \upharpoonleft \varphi)^\smile$.

Theorem 6.5 *Let L be a logic in-between K and $K_{(m)}(\cap, \cup, ^-, ^\smile, \upharpoonleft, ;^{\mathsf{pos}})$. Let Δ be a finite set of relational properties expressible in DL^*. Let N be the clausal form of $\Delta \wedge \mathrm{Def}_\Lambda \Pi(\varphi)$, where φ is any modal formula in L and Λ is defined as in Theorem 6.3. Then: (i) Any derivation from N in R^{ord} (up to redundancy) terminates in double exponential time. (ii) φ is unsatisfiable in L iff there is a refutation of N in R^{ord}.*

This theorem cannot be strengthened further by removing the restriction on compositions. From the undecidability result of the equational theory of Boolean algebras with composition in [49] it follows that allowing arbitrary occurrences of composition leads to undecidability.

Theorem 6.6 *The satisfiability problem in every logic in-between $K_{(m)}(\cap, \cup, ^-, ;)$ and $K_{(m)}(\cap, \cup, ^-, ^\smile, \upharpoonleft, ;)$ is undecidable.*

Theorem 6.7 *Every logic in-between $K_{(m)}(\cap, \cup, ^-)$ and $K_{(m)}(\cap, \cup, ^-, ^\smile, \upharpoonleft)$ is NEXPTIME-complete.*

Proof. A consequence of the NEXPTIME-completeness of the satisfiability of Boolean modal logic and FO^2 formulae [32, 51]. □

Det	$[\beta]p \to \langle\gamma\rangle p$	$\forall x \exists y\, (R_\beta(x,y) \wedge R_\gamma(x,y))$
Sym	$\langle\alpha\rangle[\beta]p \to p$	$\forall x \forall y\, (R_\alpha(x,y) \to R_\beta(y,x))$
Gr	$[\beta]p \to [\alpha]p$	$\forall x \forall y\, (R_\alpha(x,y) \to R_\beta(x,y))$
Conf	$\langle\alpha\rangle[\beta]p \to \langle\gamma\rangle p$	$\forall x \forall y\, (R_\alpha(x,y) \to \exists z\, (R_\alpha(x,z) \wedge R_\beta(z,y)))$

Figure 4: Modal axioms and their correspondence properties

From Theorem 6.5 we can obtain some decidability results for propositional modal logics. In the following let α, β and γ denote either a relational variable or a relational formula built from relational variables using disjunction and composition. Let Σ be a set modal formulae in the language of multi-modal $K_{(m)}$ and let $K_{(m)}\Sigma$ be the extension of $K_{(m)}$ closed under the formulae in Σ. For example, the axiom schema listed in Figure 4 determine classes of logics considered in Catach [9] and Baldoni [5].

Theorem 6.8 *Let Σ be any finite set of instances of formulae in Figure 4, and let Δ^Σ be the set of associated first-order properties as specified in Figure 4. Then: For any modal formula φ, φ is satisfiable in $K_{(m)}\Sigma$ iff $\Delta^\Sigma \wedge \Pi(\varphi)$ is first-order satisfiable.*

Proof. By noting that disjunction and composition in the relational parameters of the modal operators can be normalised away, it is not difficult to see that all formulae in Figure 4 are Sahlqvist formulae. Using the SCAN algorithm [19] one can prove the properties associated with the modal formulae in Figure 4 are in fact their correspondence properties. Thus, the theorem follows from the well-known Sahlqvist Theorem [68]. □

This theorem is also an easy consequence of a more general theorem by Catach [9].

Theorem 6.9 *Let Σ be any finite set of instances of formulae in Figure 4, with the restriction that in each case α is a relational formula built from relational variables and disjunction only, while β and γ denote either a relational variable or a relational formula built from relational variables using disjunction and composition. Then, the satisfiability problem in $K_{(m)}\Sigma$ is decidable, and it can be decided by a resolution procedure based on the translation into DL* and any ordering refinement compatible with $>_d$.*

Proof. The restriction that α is a relational formula built from relational disjunction only enures that relational composition occurs only positively in the first-order correspondence properties for the axioms in Σ. This implies all correspondence properties can be formulated in *DL**. The result then follows by Theorem 6.4. □

Finally, we observe that decidability is preserved if a relational property ψ in the theory Δ for a logic in-between $K_{(m)}$ and $K_{(m)}(\cap, \cup, ^-, ^\smile, \upharpoonright, ;^{\mathsf{pos}})$ may also include binary literals of the form $(\neg)R(x,x)$ or $(\neg)R(y,y)$, if x and y are the two universally quantified variables in the property ψ. An example is $\forall xy\, (R_1(x,y) \to R_2(x,x) \vee R_3(y,y))$. Although the clausal form is not an *DL** clause it can easily be transformed into a set of *DL** clauses using the renaming techniques used in Hustadt and Schmidt [43] for deciding Maslov's dual class K by ordered resolution.

7 Decision procedures using selection-based resolution

$K_{(m)}(\cap, \cup, \smile)$ and logics below it have the property that they can be decided by a refinement of resolution which is defined solely by a selection function of negative literals [13, 45]. The transformation to clausal form is based on the standard translation and we use the same structural transformation as described in Section 4, except that Def_Λ introduces the same symbol for variant subformulae with the same polarity. For simplicity it is assumed that φ is in negation normal form, that is, in every subformula of the form $\neg\psi$, ψ is a propositional variable. As a consequence all occurrences of non-atomic subformulae of φ' with one free variable have positive polarity. This means that $\mathrm{Def}_\lambda(\varphi') = \mathrm{Def}^+_\lambda(\varphi')$ for the positions λ associated with these occurrences. But subformulae corresponding to relational formulae (subformulae with two free variables) can occur both positively and negatively. For these Def_Λ introduces one symbol for all variant occurrences of subformulae corresponding to non-atomic relational subformulae with positive polarity and a different symbol for all variant occurrences with negative polarity. For example, Def_Λ maps $[\alpha]\langle\alpha\rangle p$ with $\alpha = r_1 \wedge r_2$ to the conjunction of the following formulae.

$$\begin{aligned}
&\exists x\, Q_{[\alpha]\langle\alpha\rangle p}(x)\\
&\forall x\, (Q_{[\alpha]\langle\alpha\rangle p}(x) \rightarrow \forall y\, (Q^n_\alpha(x,y) \rightarrow Q_{\langle\alpha\rangle p}(y)))\\
&\forall x\, (Q_{\langle\alpha\rangle p}(x) \rightarrow \exists y\, (Q^p_\alpha(x,y) \wedge P(y)))\\
&\forall xy\, (Q^p_\alpha(x,y) \rightarrow (R_1(x,y) \wedge R_2(x,y)))\\
&\forall xy\, ((R_1(x,y) \wedge R_2(x,y)) \rightarrow Q^n_\alpha(x,y)).
\end{aligned}$$

The symbol Q^n_α (resp. Q^p_α) is associated with the negative (resp. positive) occurrence of α.

Subsequently, introduced predicate symbols are denoted by Q_ψ and Q^p_α or Q^n_α, where Q_ψ represents an occurrence of a modal subformula ψ and Q^p_α (Q^n_α) represents a positive (negative) occurrence of a relational subformula α. Let

$\mathcal{P}(s)$ denote some literal in $\{P_i(s), Q_\psi(s)\}_{i,\psi}$, and let

$\mathcal{R}(s,t)$ denote some literal in $\{R_j(s,t), R_j(t,s), Q^{p/n}_\alpha(s,t), Q^{p/n}_\alpha(t,s)\}_{j,\alpha}$.

Note two occurrences of $\mathcal{P}(s)$ or $\mathcal{R}(s,t)$ need not be identical. For example, $\neg Q_\psi(x) \vee P_i(x) \vee Q_\chi(x)$ is an instance of $\neg Q_\psi(x) \vee \mathcal{P}(x) \vee \mathcal{P}(x)$, while $\neg Q_\psi(x) \vee \neg R_j(y,x) \vee Q_\chi(y)$ and $\neg Q_\psi(x) \vee \neg Q^n_\alpha(x,y) \vee Q_\chi(y)$ are instances of $\neg Q_\psi(x) \vee \neg\mathcal{R}(x,y) \vee \mathcal{P}(y)$.

All input clauses have one of the forms described in Figure 5 [13, 45]. The literals marked with $^+$ are selected in the clauses by the specific selection function we use.

The calculus is based on maximal selection of negative literals. This means the selection function selects exactly the set of all negative literals in any non-positive clause. An ordering refinement is optional. In this case, the ordered resolution rule of R can be replaced by the following rule.

$$
\begin{array}{ll}
\quad\mathcal{P}(a) & \\
\neg Q_\psi(x)^+ \vee \neg P_i(x)^+ & \text{if } \psi = \neg p_i \\
\neg Q_\psi(x)^+ \vee \mathcal{P}(x)\ [\vee\, \mathcal{P}(x)] & \text{if } \psi = \phi_1 \wedge[\vee]\ \phi_2 \\
\neg Q_\psi(x)^+ \vee \neg \mathcal{R}(x,y)^+\ [\vee\, \mathcal{P}(y)] & \text{if } \psi = [\alpha]\phi\ [\psi = [\alpha]\bot] \\
\neg Q_\psi(x)^+ \vee \mathcal{P}(f(x)) & \multirow{2}{*}{\text{if } \psi = \langle\alpha\rangle\phi} \\
\neg Q_\psi(x)^+ \vee \mathcal{R}(x, f(x)) & \\
\neg Q^p_\alpha(x,y)^+ \vee \mathcal{R}(x,y)\ [\vee\, \mathcal{R}(x,y)] & \text{if } \alpha = \beta_1 \wedge[\vee]\ \beta_2 \text{ has pos. polarity} \\
\quad Q^n_\alpha(x,y) \vee \neg\mathcal{R}(x,y)^+\ [\vee\, \neg\mathcal{R}(x,y)^+] & \text{if } \alpha = \beta_1 \wedge[\vee]\ \beta_2 \text{ has neg. polarity}
\end{array}
$$

Figure 5: Schematic clausal forms for $K_{(m)}(\cap, \cup, \smallsmile)$

Resolution with maximal selection:

$$
\frac{C_1 \vee A_1 \quad \cdots \quad C_n \vee A_n \quad \neg A_{n+1} \vee \ldots \vee \neg A_{2n} \vee D}{(C_1 \vee \ldots \vee C_n \vee D)\sigma}
$$

provided for any $1 \leq i \leq n$, (i) σ is the most general unifier of A_i and A_{n+i}, (ii) $C_i \vee A_i$ and D are positive clauses, (iii) no A_i occurs in C_i, and (iv) $A_i, \neg A_{n+i}$ are selected. The *negative premise* is $\neg A_{n+1} \vee \ldots \vee \neg A_{2n} \vee D$ and the other premises are the *positive premises*. The literals A_i and A_{n+i} are the *eligible literals*.

Let $\mathsf{R}^{\mathsf{hyp}}$ be the calculus based on maximal selection and no ordering. This means the rules are the above resolution rule, positive unordered factoring and splitting. The normalisation function is not needed (but could of course be added without losing completeness), that is, we asumme NORM is the identity mapping. For simplification tautology deletion is used. All derivations in $\mathsf{R}^{\mathsf{hyp}}$ are generated by strategies in which no application of the resolution or factoring with identical premises and identical consequence may occur twice on the same path in any derivation. In addition, deletion rules, splitting, and the deduction rules are applied in this order, except that splitting is not applied to clauses which contain a selected literal.

As all non-unit clauses of a typical input set contain a selected literal all definitional clauses can only be used as negative premises of resolution steps. To begin with there is only one candidate for a positive premise, namely, the ground unit clause $Q_\varphi(a)$ representing the input formula φ. Inferences with such ground unary unit clauses produce ground clauses consisting of positive literals only, which will be split into ground unit clauses.

Lemma 7.1 ([45]) *Maximally split (non-empty) inferred clauses have one of two forms: $\mathcal{P}(s)$, or $\mathcal{R}(s, f(s))$, where s is a ground term.*

In general, s will be a nested non-constant functional ground term, which is typically avoided in resolution decision procedures based on an ordering refinement, because in most situations nesting causes unbounded computations. However, it can be shown that for the class of clauses under consideration any derived clause is smaller than its positive parent clauses with respect to a well-founded ordering which reflects the structure of the formula.

Theorem 7.2 ([45]) *Let L be a logic in-between K and $K_{(m)}(\cap, \cup, ^-, \smile)$. Let φ be any L-formula and let N be the clausal form of $\mathrm{Def}_\Lambda\Pi(\varphi)$. Then: (i) Any R^{hyp}-derivation from N terminates. (ii) φ is unsatisfiable in L iff there is a refutation of N by R^{hyp}.*

Theorem 7.3 ([13]) *For any logic in-between K and $K_{(m)}(\cap, \cup, \smile)$, the space complexity for testing the satisfiability of a modal formulae φ with R^{hyp} is bounded by $O(nd^m)$, where n is the number of symbols in φ, d is the number of different diamond subformulae in φ, and m is the modal depth of φ.*[7]

Formulae in $K_{(m)}(\cap, \cup, \smile)$ translate by Π into the guarded fragment, while there are formulae in $K_{(m)}(\cap, \cup, ^-, \smile)$ which do not [13]. It is not difficult to see that formulae in $K_{(m)}(\cap, \cup, \smile)$ are in fact translated into the subfragment *GF1*$^-$, introduced by Lutz, Sattler and Tobies [52]. In contrast to the guarded fragment, *GF1*$^-$ permits the development of space-optimal decision procedures [26, 52]. Under the assumption that either (i) there is a bound on the arity of predicate symbols in *GF1*$^-$ formulae, or (ii) that each subformula of a *GF1*$^-$ formula has a bounded number of free variables, the satisfiability problem of *GF1*$^-$ is the same as for $K_{(m)}$ [52]. Thus, we can conclude:

Theorem 7.4 *The computational complexity of the satisfiability problem of any modal logic in-between $K_{(m)}$ and $K_{(m)}(\cap, \cup, \smile)$ is PSPACE-complete.*

In [26] it is shown that R^{hyp} can be implemented as a modification of the main procedure of a standard (saturation based) first-order theorem prover with splitting (e.g. SPASS) to provide a space optimal decision procedure for *GF1*$^-$. A direct consequence is the following.

Theorem 7.5 *R^{hyp} can be turned into a polynomial space resolution decision procedure for logics in-between $K_{(m)}$ and $K_{(m)}(\cap, \cup, \smile)$.*

So far in this section we have considered only logics with empty relational theory. It is natural to try and strengthen the results obtained. We might ask whether the results can be generalised, and if it is indeed possible, to try and determine for which theories the above results can be generalised. Generalisations of Theorem 7.2 have been considered in [13, 45]. We quote here a generalised theorem established in [13].

Theorem 7.6 ([13]) *Let L be a logic in-between K and $K_{(m)}(\cap, \cup, \smile)$. Let Δ be a finite R^{hyp}-saturated set of clauses consisting of two kinds of split components.*

[7] By definition the *modal depth* of a formula φ is the maximal nesting of modal operators $\langle\alpha\rangle$ or $[\alpha]$ in φ.

1. *Clauses with at most two free variables, which are built from finitely many binary predicate symbols R_j, no function symbols, and containing at least one guard literal (that is, this literal is negative and includes all the variables of the clause).*

2. *Clauses built from one variable, finitely many function symbols (including constants), and finitely many binary predicate symbols R_j, with the restriction that (a) the argument multisets of all non-ground literals coincide, and (b) each literal which contains a constant is ground.*

Suppose φ is an L-formula and N is the clausal form of $\mathrm{Def}_\Lambda\Pi(\varphi)$. *Then: (i) Any R^{hyp}-derivation from $N \cup \Delta$ terminates. (ii) φ is unsatisfiable in $L\Delta$ iff there is a refutation of $N \cup \Delta$ by R^{hyp}.*

Relational frame properties covered by this result include reflexivity, irreflexivity, seriality, symmetry, inclusions among relations, for example, $R_1 \subseteq R_2$ or $R_1 \subseteq (R_2^{\smile} \cap R_3)$, as well as, for example, $\forall x \exists y\, \neg R(x,y)$, $\forall x \exists y\, (R(x,y) \vee R(y,x))$, or $\forall xy\, (R(x,y) \rightarrow R(x,x))$. Of the properties in Figure 4 the properties Det, Sym and Gr are covered, provided that the relational parameters α, β and γ are formed from relational variables and disjunction. Thus, familiar logics covered by the above results include *KT*, *KD*, *KB*, *KTB*, and *KDB*, but also the basic tense logic K_t.

The results of this section also cover the corresponding description logics, for example, the basic description logic $\mathcal{ALC}$ possibly extended with role conjunction, role disjunction and inverse roles. Acyclic TBox statements, and both concept and role ABox statements are also in the scope of the last theorem.

8 Simulating tableaux

Selection refinements of resolution (and hyperresolution) are closely related to standard modal tableau calculi and description logic systems [13, 17, 44, 45, 47]. In this section investigate simulation relationships between the selection-based resolution procedure R^{hyp} and Massacci's single-step prefixed tableau calculi [53].

There are three notions of simulation [13]: polynomial simulation of derivations, polynomial simulation of search, and step-wise simulation. By definition, a proof system $\mathcal{A}$ *p-simulates (polynomially simulates) derivations* of a proof system $\mathcal{B}$ iff there is a function g, computable in polynomial time, which maps derivations in $\mathcal{B}$ for any given formula φ, to derivations in $\mathcal{A}$ for φ. A system $\mathcal{A}$ *p-simulates search* of a system $\mathcal{B}$ iff there is a polynomial function g such that for any formula φ, g maps derivations from φ in $\mathcal{A}$ to derivations from φ in $\mathcal{B}$. The first notion generalises the notion of p-simulation found in [10], who are only concerned with the p-simulation of proofs (that is, successful derivations leading to a proof). Simulation of search is a relationship in the opposite direction. It implies that $\mathcal{A}$ does not perform any inference steps for which no corresponding inference steps exist in $\mathcal{B}$. To show that $\mathcal{A}$ p-simulates proofs or derivations of $\mathcal{B}$ it is sufficient to prove that for every formula φ and every derivation $D_{\mathcal{B}}$ of φ in $\mathcal{B}$, there exists a derivation $D_{\mathcal{A}}$ of φ in $\mathcal{A}$ such that the number of applications of inference rules

$$(\bot)\ \frac{\sigma : \psi,\ \sigma : \neg\psi}{\sigma : \bot} \qquad (\wedge)\ \frac{\sigma : \psi \wedge \phi}{\sigma : \psi,\ \sigma : \phi} \qquad (\vee)\ \frac{\sigma : \psi \vee \phi}{\sigma : \psi \mid \sigma : \phi}$$

$$(\Diamond)\ \frac{\sigma : \Diamond\psi}{\sigma.n : \psi} \text{ with } \sigma.n \text{ new to the current branch}$$

$$(\Box)\ \frac{\sigma : \Box\psi}{\sigma.n : \psi} \qquad (D)\ \frac{\sigma : \Box\psi}{\sigma : \Diamond\psi} \qquad (T)\ \frac{\sigma : \Box\psi}{\sigma : \psi}$$

$$(B)\ \frac{\sigma.n : \Box\psi}{\sigma : \psi} \qquad (4)\ \frac{\sigma : \Box\psi}{\sigma.n : \Box\psi} \qquad (4^r)\ \frac{\sigma.n : \Box\psi}{\sigma : \Box\psi}$$

$$(4^d)\ \frac{\sigma.n : \Box\psi}{\sigma.n.m : \Box\psi} \qquad (5)\ \frac{1.n : \Box\psi}{1 : \Box\Box\psi}$$

Figure 6: Single step prefixed tableau expansion rules for subsystems of *S5*.

K:	(K)	$K5$:	$(K), (4^r), (4^d), (5)$	$KD5$:	$(K), (D), (4^r), (4^d), (5)$
KD:	$(K), (D)$	KDB:	$(K), (D), (B)$	$KB4$:	$(K), (B), (4), (4^r)$
KT:	$(K), (T)$	$KD4$:	$(K), (D), (4)$	$K45$:	$(K), (4), (4^r), (4^d)$
KB:	$(K), (B)$	KTB:	$(K), (T), (B)$	$KD45$:	$(K), (D), (4), (4^r), (4^d)$
$K4$:	$(K), (4)$	$S4$:	$(K), (T), (4)$	$S5$:	$(K), (T), (4), (4^r)$

Figure 7: Tableau calculi for subsystems of *S5*. (K) denotes the sequence of rules $(\bot), (\wedge), (\vee), (\Diamond), (\Box)$.

in $D_{\mathcal{A}}$ is polynomially bounded by the number of applications of inference rules in $D_{\mathcal{B}}$. This can be achieved by showing that there exists a number n such that each application of an inference rule in $D_{\mathcal{A}}$ corresponds to at most n applications of inference rules in $D_{\mathcal{B}}$. It follows that the length of derivation $D_{\mathcal{B}}$ is polynomially bounded by the length of $D_{\mathcal{A}}$. This is known as a *step-wise simulation* of $\mathcal{B}$ by $\mathcal{A}$ [15]. Note that a step-wise simulation is independent of whether the considered derivations are proofs or not.

The single-step prefixed tableau calculi of Massacci [53] for subsystems of *S5* are defined by Figures 6 and 7. The basic entities are formulae labelled with prefixes. A labelled (prefixed) formula has the form $\sigma : \varphi$, where σ is a sequence of positive integers and φ is a modal formula. σ represents a world in which φ is true. Tableau derivations have a tree structure and begin with the formula, $1 : \varphi$ in the root node. Successor nodes are then constructed by the application of the expansion rules in Figure 6. The prefixes in the expansion rules, except for $\sigma.n$ of the $(\Diamond)$-rule, are assumed to be present on the current branch.

Theorem 8.1 ([53]) *Let $\Sigma \subseteq \{D, T, B, 4, 5\}$. A formula φ is satisfiable in a logic $K\Sigma$ iff a tableau containing a branch $\mathcal{B}$ can be constructed by the tableau calculus for $K\Sigma$ such that $\mathcal{B}$ does not contain the falsum and further rule applications are redundant.*

Theorem 8.2 ([47]) *Let $\Sigma \subseteq \{D, T, B, 4, 5\}$. There is a p-simulation of single step prefix tableau derivations for $K\Sigma$ using $\mathsf{R}^{\mathsf{hyp}}$.*

The proof exploits the step-wise simulation of tableau inference steps by resolution inference steps where the theories are given by the clausal form of the conjunction of the first-order correspondence properties of the axioms. For the modal logics $K\Sigma$ with $\Sigma \subseteq \{D, T, B\}$ simulation in the other direction can also be proved.

Theorem 8.3 ([47]) *$\mathsf{R}^{\mathsf{hyp}}$ p-simulates search in single step prefix tableaux for $K\Sigma$ with $\Sigma \subseteq \{D, T, B\}$.*

This is a consequence of a near bisimulation between the tableau derivations and $\mathsf{R}^{\mathsf{hyp}}$ derivations for the logics under consideration. If factoring rules are added to the single step prefix tableau calculus then this calculus can also p-simulate $\mathsf{R}^{\mathsf{hyp}}$ derivations.

Because $\mathsf{R}^{\mathsf{hyp}}$ does not terminate in the presence of the transitivity clause or Euclideanness, Theorem 8.3 does not extend to transitive and Euclidean modal logics. For 4 and 5 termination in single step prefixed tableaux is ensured by a loop checking mechanism [53]. Once a loop is detected in a branch no further rules are applied. In $\mathsf{R}^{\mathsf{hyp}}$ further inference steps will be performed. To prevent this we would have to provide a means by which the resolution procedure can recognise the redundancy of further inference steps. This can be realised with a blocking inference rule, used in [44], which has an effect similar to loop checking. Using soft typing described in [22] might provide an alternative solution.

Similar simulation results can be obtained for other forms of modal tableau calculi, including caluli with implicit or explicit accessibility relation and analytic modal KE tableaux, e.g. [41, 53], or even sequent proof systems. Simulation results of tableau calculi for description logics by resolution can be found in Hustadt and Schmidt [44, 45].

9 Developing tableaux via resolution

The simulation of tableau calculi in the resolution framework can be exploited for systematically developing sound and complete tableau rule systems. For instance, De Nivelle et al [13] show how a tableau system for $K_{(m)}(\cap, \cup, \smile)$ can be extracted from the resolution method described in Section 7. The idea is to express a $\mathsf{R}^{\mathsf{hyp}}$ resolution inference step by a tableau rule, or if this is not possible, as is the case for conjunctive subformulae, to express a group of $\mathsf{R}^{\mathsf{hyp}}$ resolution inference steps as a tableau rule.

A *tableau* is a finitely branching tree whose nodes are sets of labelled formulae. Given that φ is a formula to be tested for satisfiability the root node is the set $\{a : \varphi\}$. Successor nodes are constructed in accordance with a set of expansion rules. A rule $\frac{X}{X_1 \mid \ldots \mid X_n}$ fires for a selected formula F in a node if F is an instance of the numerator X, or more generally, F together with other formulae in the node are instances of the formulae in X. n successor nodes are created which contain the formulae of the current node and the appropriate instances of X_i. It is assumed that no rule is applied twice to the same instance of the numerator.

$$(\bot)\ \frac{s:\psi,\ s:\neg\psi}{s:\bot} \qquad (\wedge)\ \frac{s:\psi\wedge\phi}{s:\psi,\ s:\phi} \qquad (\vee)\ \frac{s:\psi\vee\phi}{s:\psi \mid s:\phi}$$

$$(\Diamond)\ \frac{s:\langle\alpha\rangle\psi}{(s,t):\alpha,\ t:\psi}\ \text{with } t \text{ new to the branch} \qquad (\Box)\ \frac{(s,t):\alpha,\ s:[\alpha]\psi}{t:\psi}$$

$$(\smile)\ \frac{(s,t):\alpha^{\smile}}{(t,s):\alpha} \qquad (\wedge^r)\ \frac{(s,t):\alpha\wedge\beta}{(s,t):\alpha,\ (s,t):\beta} \qquad (\vee^r)\ \frac{(s,t):\alpha\vee\beta}{(s,t):\alpha \mid (s,t):\beta}$$

$$(\smile_I)\ \frac{(t,s):\alpha}{(s,t):\alpha^{\smile}} \qquad (\wedge^r_I)\ \frac{(s,t):\alpha,\ (s,t):\beta}{(s,t):\alpha\wedge\beta} \qquad (\vee^r_I)\ \frac{(s,t):\alpha}{(s,t):\alpha\vee\beta}$$

Figure 8: Tableau expansion rules for $K_{(m)}(\cap,\cup,\smile)$. For the rules $(\smile_I)$, $(\wedge^r_I)$ and $(\vee^r_I)$ the side conditions are that the formulae in the denumerator, i.e. $\alpha^{\smile}$, $\alpha\wedge\beta$ or $\alpha\vee\beta$, occur as subformulae of the parameter γ of a box formula $s:[\gamma]\psi$ on the current branch.

Assume that φ is a formula in negation normal form. Recall, the inference in R^{hyp} starts with an inference with the only positive clause $Q_\varphi(a)$. Accordingly, the root of the tableau is given by $\{a:\varphi\}$. Subsequent R^{hyp} inference steps can be translated more or less directly into the tableau expansion rules listed in Figure 8 (for details see [13]). The rules for $K_{(m)}(\cap,\cup,\smile)$ include the clash rule $(\bot)$, seven 'elimination' rules $(\wedge)$, $(\vee)$, $(\Diamond)$, $(\Box)$, $(\smile)$, $(\wedge^r)$, and $(\vee^r)$ for positive occurrences of subformulae, and three 'introduction' rules $(\smile_I)$, $(\wedge^r_I)$ and $(\vee^r_I)$ for negative occurrences of subformulae. The side conditions for the introduction rules ensure that formulae are not introduced unnecessarily. Conjunction and disjunction are assumed to be associative and commutative operations. Only the disjunction rules are don't know nondeterministic and require the use of backtracking. For any logic L in-between K and $K_{(m)}(\cap,\cup,\smile)$ the expansion rules are given by appropriate subsets.

Unnecessary duplication and superfluous inferences can be kept to a minimum by adopting a notion of redundancy which is in the spirit of Bachmair and Ganzinger [2]. A labelled formula F is redundant in a node if the node contains labelled formulae $F_1,\dots,F_n$ (for $n\geq 0$) which are smaller than F and $\models_L (F_1\wedge\ldots\wedge F_n)\rightarrow F$. In this context a formula ψ is smaller than a formula ϕ if ψ is a subformula of ϕ, but a more general definition based on an admissible ordering in the sense of [2, 3] may be chosen. The application of a rule is redundant if its premise(s) or its conclusion(s) is (are) redundant in the current node. For example, for any s, $s:\top$ is redundant, and if a node includes $s:\psi$ and $s:\psi\vee\phi$, then the $(\vee)$ rule need not be applied, and no new branches are introduced.

Theorem 9.1 ([13]) *A formula φ is satisfiable in $K_{(m)}(\cap,\cup,\smile)$ iff a tableau containing a branch $\mathcal{B}$ can be constructed with the rules of Figure 8 such that $\mathcal{B}$ does not contain falsum ($s:\bot$ for some s) and each rule application is redundant.*

Corollary 9.2 ([13]) *The appropriate subsets of the rules from Figure 8 provide sound, complete and terminating tableau calculi for logics in-between K and $K_{(m)}(\cap,\cup,\smile)$.*

10 Generating Herbrand models

A problem closely related to the satisfiability problem is the problem of generating (counter-)models. It is well-known that hyperresolution can be employed with dual purpose, namely, as a reasoning method and a Herbrand model builder [17]. Thus, in this section we briefly discuss the use of $\mathsf{R}^{\mathsf{hyp}}$ as a procedure for automatically constructing Herbrand models for extended modal logics. The results are actually consequences of properties of classes of range restricted clause sets. A clause C is said to be *range restricted* iff the set of variables of the positive part of C is a subset of the set of variables of the negative part of C. A clause set is range restricted iff it contains only range restricted clauses. This means that a positive clause is range restricted only if it is a ground clause.

A *Herbrand interpretation* is a set of ground atoms. By definition a ground atom A is *true* in an interpretation H if $A \in H$ and it is *false* in H if $A \notin H$, $\top$ is true in all interpretations and $\bot$ is false in all interpretations. A literal $\neg A$ is true in H iff A is false in H. A conjunction of two ground atoms A and B is true in an interpretation H iff both A and B are true in H and respectively, a disjunction of ground atoms is true in H iff at least one of A or B is true in the interpretation. A clause C is true in H iff for all ground substitutions σ there is a literal L in $C\sigma$ which is true in H. A set N of clauses is true in H iff all clauses in N are true in H. If a set N of clauses is true in an interpretation H then H is referred to as a *Herbrand model* of N.

For range restricted clause sets the procedure $\mathsf{R}^{\mathsf{hyp}}$ implicitly generates Herbrand models [8, 26, 30]. For a class of solvable range restricted clauses, if $\mathsf{R}^{\mathsf{hyp}}$ terminates on a clause set N without having produced a refutation then a model can be extracted from any complete, open branch in the derivation. The model is given by the set of ground unit clauses in the limit of the branch, i.e. the clause set at the leaf of the branch.

Theorem 10.1 ([47]) *Let N be the clausal form of a $K_{(m)}(\cap, \cup, \smile)$ formula φ (as defined in Section 7), and let N_∞ be the limit of an arbitrary branch $\mathcal{B}$ in a $\mathsf{R}^{\mathsf{hyp}}$ derivation tree with root N. Let $[\![\mathcal{B}]\!]$ be the set of positive ground unit clauses in N_∞. If N_∞ does not contain the empty clause, then $[\![\mathcal{B}]\!]$ is a finite (Herbrand) model of N.*

Now a modal model $\mathcal{M} = (W, R, \iota)$ can be easily constructed from $[\![\mathcal{B}]\!]$ for φ. Essentially, the set of worlds is defined by the set of ground terms occurring in $[\![\mathcal{B}]\!]$. The interpretation of relational formulae is determined by the set of R_i literals in $[\![\mathcal{B}]\!]$. For any R_i, if $R_i(s,t)$ is in $[\![\mathcal{B}]\!]$ then $(s,t) \in R(r_i)$, which can be extended to a homomorphism for complex relational formulae. The interpretation of modal formulae can be defined similarly. For any unary literal $P_i(s)$ (resp. $Q_\psi(s)$) in $[\![\mathcal{B}]\!]$, $s \in \iota(p_i)$ (resp. $s \in \iota(\psi)$), that is, p_i (resp. ψ) is true in the world s. This is homomorphically extended as expected. Consequently:

Theorem 10.2 ([13, 47]) *Let L and Δ be as in Theorem 7.6. For any modal formula satisfiable in $L\Delta$ a finite modal model can be effectively constructed on the basis of $\mathsf{R}^{\mathsf{hyp}}$.*

Corollary 10.3 ([13, 47]) *Let L and Δ be as in Theorem 7.6. Then, $L\Delta$ has the finite model property.*

These results extend also to methods closely related to $\mathsf{R}^{\mathsf{hyp}}$ such as the tableau calculus introduced in the previous section. For example:

Corollary 10.4 ([13]) *If L is a logic in-between K and $K_{(m)}(\cap, \cup, \smile)$, and φ is satisfiable in L then a finite modal model can be effectively constructed on the basis of the tableau calculus for L given by the appropriate subset of rules from Figure 8.*

11 Mechanisation

There are a number of first-order theorem provers that implement the Bachmair-Ganzinger framework of resolution and equality reasoning [3]. These include E [72, 73], SATURATE [23], SPASS [74, 75] VAMPIRE [66]. The theorem prover OTTER [54] also implements ordered resolution and hyperresolution. These provers are sophisticated programs which have been developed over many years. Of these provers, SPASS forms the basis of the theorem prover MSPASS which has been used to study the practical properties of $\mathsf{R}^{\mathsf{ord}}$ and $\mathsf{R}^{\mathsf{hyp}}$ for automating modal logic reasoning [42, 46, 47, 48, 69]. These studies have been mainly for basic multi-modal logic. Furthermore, with one exception, a variation of the optimised functional translation method was used, since MSPASS has shown a better performance for the optimised functional translation method compared to the relational translation method on the benchmark problems used in the studies. However, according to current knowledge the practical scope of the optimised functional translation method is limited to modal logics with K or KD modalities.

The main difference between SPASS and MSPASS is that MSPASS accepts also modal logic, description logic and relational formulae as input. Modal formulae and description logic formulae are built from a vocabulary of propositional symbols of two disjoint types, namely, propositional (Boolean or concept) and relational (role). The repertoire of logical constructs includes: (i) the standard Boolean operators on both propositional and relational formulae, (ii) multiple modal operators, permitting complex relational parameters, i.e. $\langle \_ \rangle$ and $[\_]$, as well as the domain and range operators, (iii) the relational operators, composition, relative sum, converse, identity, diversity, and (iv) the test operator of PDL, domain restriction and range restriction. MSPASS supports non-logical axioms which are true in every possible worlds (that is, global satisfiability, or generalised concept or role terminological axioms for description logics). In addition, it is possible to specify additional frame properties, or any other first-order restrictions on the translated formulae.

Among the mentioned theorem provers the following features make (M)SPASS the most well-suited theorem prover for our purposes. (M)SPASS includes a fast converter of first-order logic formulae in clausal form with special features such as optimised and strong Skolemisation, and an improved implementation of renaming [58]. (M)SPASS supports splitting and branch condensing (branch condensing resembles branch pruning or backjumping). Ordered inference, splitting, and condensing are of particular importance concerning the performance for satisfiable formulae, and for randomly generated formulae unit propagation and branch condensing are important as well.

Mechanisation of the approaches described in this paper is not difficult. All that is needed is to select a correct set of flag settings to turn MSPASS into implementations of a particular combination of a translation and $\mathsf{R}^{\mathsf{ord}}$ or $\mathsf{R}^{\mathsf{hyp}}$.

Although MSPASS does not provide a decision procedure for all the modal logics one might be interested in, for example, *PDL* or graded modal logic are not supported, an attractive feature of MSPASS is the possibility to specify arbitrary first-order theories. Anything which can be encoded into first-order logic with equality can be expressed with MSPASS. This allows for its use as a flexible tool for the investigation of combinations of interacting non-classical logics or description logics, which have not been been studied in depth before, and in particular, which have not been anticipated by the implementors. In this context it is useful that, on termination, MSPASS does not only produce a 'yes'/'no' answer, but it also outputs a proof or a saturated set of clauses, depending on whether input problem is unsatisfiable or satisfiable. A finite saturated set of clauses provides a characterisation of a class of models for the input problem. In the case $\mathsf{R}^{\mathsf{hyp}}$ is used the generated clauses define a Herbrand model.

12 Topics not covered

The combination of translation and first-order inference methods provides a powerful and versatile approach for studying and mechanising reasoning, model generation and other aspects of modal logic. Due to space restrictions we had to be selective in our choice of topics covered in this overview. Some important topics omitted in this overview include the following.

Non-standard translation approaches. Non-standard translation methods include reductions derived from the functional semantics of normal modal logics with unparameterised modalities, namely the functional translation [1, 35, 59], the optimised functional translation [35, 63, 76] and the semi-functional translation [56]. Surveys of the different translation methods of modal logics and other non-classical logics are [60, 61, 62]. These non-standard translation approaches are all implemented in MSPASS [46]. Experience shows that the performance of first-order theorem provers is best for a variation of the optimised functional translation [42].

Deciding modal logics with transitive modalities. To decide extensions of *K4* one possibility is to modify the calculus and add ordered chaining rules for transitive relations [21]. Another possibility is to use the resolution procedures described in this paper, or elsewhere, for example [70], but block further inferences with clauses containing terms in which the level of nesting exceeds a pre-computed term depth bound. In practice this solution is rather poor, as are solutions encoding *K4* or *S4* problems in *K* or *KT*. A possibility largely unexplored to date is to turn the resolution-tableaux relationship around and develop resolution decision procedures from tableau decision procedures.

Mechanising correspondence theory. Computing the first-order correspondence property, if it exists, for a modal formula amounts to the elimination of the universal monadic second-order quantifiers expressing the validity in a frame of that formula or, equivalently, the elimination of the existential monadic second-order quantifiers expressing the satisfiability of the formula. One of the best known algorithms for elimination of existential second-order quantifiers is SCAN, developed by Ohlbach and Gabbay [19]. The SCAN algorithm is based on constraint resolution and was implemented by Engel [16] as an extension of the OTTER theorem prover. The SCAN algorithm is known to be sound, meaning that whenever the algorithm terminates successfully the resulting formula is equivalent to the original formula. Unfortunately, it is provably impossible for such a reduction (by SCAN or any other method) to be always successful, even if there is a simpler equivalent formula for a second-order logic formula. However, even though SCAN cannot be complete in this general sense, SCAN has been shown to be useful for computing first-order frame correspondents for modal axiom schemata. It is known that SCAN can compute the frame correspondence properties for very many well-known axioms such as T, 4, 5, etcetera.

But SCAN is not the only quantifier elimination algorithm, though to date it is the only algorithm based on resolution. Another algorithm is the DLS algorithm, due to Doherty, Lukaszewicz and Szalas [14] which is also suitable for computing modal correspondence properties. The DLS algorithm was implemented by Gustafsson [33]. Both SCAN and the DLS algorithm can be used remotely. An overview of these and other quantifier elimination algorithms is [57].

Generating minimal Herbrand models. In general Herbrand models are not unique and can be large. Therefore there is a need for generating minimal models. There are various approaches to generating minimal Herbrand models with hyperresolution [6, 8, 34, 55]. With a moderate extension of R^{hyp} it is possible to guarantee the generation of all and only minimal Herbrand models for any modal and description logics reducible to a decidable class of range restricted clauses. This follows from [8] and recent investigations of $GF1^-$ and the class $\mathcal{BU}$ [26, 27, 28]. An alternative approach proposed in [26, 27, 28] uses a variant of a local minimality test developed for propositional logic.

Acknowledgements

We thank Hans de Nivelle, coauthor of the survey [13] on which a part of this text is based. The work is supported by Action 274 of COST, a European cooperation in the field of Scientific and Technical Research, and research grants GR/M36700 and GR/M88761 from the UK Goverment's Engineering and Physical Sciences Research Council.

References

[1] Y. Auffray and P. Enjalbert. Modal theorem proving: An equational viewpoint. *Journal of Logic and Computation*, 2(3):247–297, 1992.

[2] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994.

[3] L. Bachmair and H. Ganzinger. Resolution theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 2, pages 19–99. Elsevier, 2001.

[4] L. Bachmair, H. Ganzinger, and U. Waldmann. Superposition with simplification as a decision procedure for the monadic class with equality. In G. Gottlob, A. Leitsch, and D. Mundici, editors, *Proceedings of the Third Kurt Gödel Colloquium (KGC'93)*, volume 713 of *Lecture Notes in Computer Science*, pages 83–96. Springer, 1993.

[5] M. Baldoni. Normal multimodal logics with interaction axioms. In D. Basin, M. D'Agostino, D. M. Gabbay, and L. Vigano, editors, *Labelled Deduction*, pages 33–57. Kluwer, 2000.

[6] P. Baumgartner, J. D. Horton, and B. Spencer. Merge path improvements for minimal model hyper tableaux. In N. V. Murray, editor, *Proceedings of the 8th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'99)*, volume 1617 of *Lecture Notes in Artificial Intelligence*, pages 51–65. Springer, 1999.

[7] T. Boy de la Tour. An optimality result for clause form translation. *Journal of Symbolic Computation*, 14:283–301, 1992.

[8] F. Bry and A. Yahya. Positive unit hyperresolution tableaux for minimal model generation. *Journal of Automated Reasoning*, 25(1):35–82, 2000.

[9] L. Catach. Normal multimodal logics. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI'88)*, pages 491–495. AAAI Press/MIT Press, 1988.

[10] S. A. Cook and R. A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979.

[11] H. de Nivelle. Splitting through new proposition symbols. In R. Nieuwenhuis and A. Voronkov, editors, *Proceedings of the 8th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2001)*, volume 2250 of *Lecture Notes in Artificial Intelligence*, pages 172–185. Springer, 2001.

[12] H. de Nivelle and M. de Rijke. Deciding the guarded frgament by resolution. Manuscript. To appear in *Journal of Symbolic Computation*, 1999.

[13] H. de Nivelle, R. A. Schmidt, and U. Hustadt. Resolution-based methods for modal logics. *Logic Journal of the IGPL*, 8(3):265–292, 2000.

[14] P. Doherty, W. Lukaszewicz, and A. Szalas. Computing circumscription revisited: A reduction algorithm. *Journal of Automated Reasoning*, 18(3):297–336, 1997.

[15] E. Eder. *Relative Complexities of First Order Calculi.* Artificial Intelligence. Vieweg, Wiesbaden, 1992.

[16] T. Engel. Quantifier elimination in second-order predicate logic. Diplomarbeit, Fachbereich Informatik, Univ. des Saarlandes, Saarbrücken, 1996.

[17] C. Fermüller, A. Leitsch, U. Hustadt, and T. Tammet. Resolution decision procedures. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume II, chapter 25, pages 1791–1849. Elsevier, 2001.

[18] C. Fermüller, A. Leitsch, T. Tammet, and N. Zamov. *Resolution Method for the Decision Problem*, volume 679 of *Lecture Notes in Computer Science.* Springer, 1993.

[19] D. M. Gabbay and H. J. Ohlbach. Quantifier elimination in second-order predicate logic. *South African Computer Journal*, 7:35–43, 1992. Also published in B. Nebel, C. Rich, W. R. Swartout, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*, pages 425–436. Morgan Kaufmann, 1992.

[20] H. Ganzinger and H. de Nivelle. A superposition decision procedure for the guarded fragment with equality. In *Proceedings of the Fourteenth Annual IEEE Symposium on Logic in Computer Science (LICS'99)*, pages 295–303. IEEE Computer Society Press, 1999.

[21] H. Ganzinger, U. Hustadt, C. Meyer, and R. A. Schmidt. A resolution-based decision procedure for extensions of K4. In M. Zakharyaschev, K. Segerberg, M. de Rijke, and H. Wansing, editors, *Advances in Modal Logic, Volume 2*, volume 119 of *Lecture Notes*, chapter 9, pages 225–246. CSLI Publications, Stanford, 2001.

[22] H. Ganzinger, C. Meyer, and C. Weidenbach. Soft typing for ordered resolution. In *Proceedings of the 14th International Conference on Automated Deduction (CADE-14)*, volume 1249 of *Lecture Notes in Artificial Intelligence*, pages 321–335, Townsville, Australia, 1997. Springer.

[23] H. Ganzinger and R. Nieuwenhuis. The Saturate system. `http://www.mpi-sb.mpg.de/SATURATE/Saturate.html`, 1994.

[24] G. Gargov and S. Passy. A note on Boolean modal logic. In P. P. Petkov, editor, *Mathematical Logic: Proceedings of the 1988 Heyting Summerschool*, pages 299–309, New York, 1990. Plenum Press.

[25] G. Gargov, S. Passy, and T. Tinchev. Modal environment for Boolean speculations. In D. Skordev, editor, *Mathematical Logic and its Applications: Proceedings of the 1986 Gödel Conference*, pages 253–263. Plenum Press, 1987.

[26] L. Georgieva, U. Hustadt, and R. A. Schmidt. Computational space efficiency and minimal model generation for guarded formulae. In R. Nieuwenhuis and A. Voronkov, editors, *Proceedings of the 8th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2001)*, volume 2250 of *Lecture Notes in Artificial Intelligence*, pages 85–99. Springer, 2001.

[27] L. Georgieva, U. Hustadt, and R. A. Schmidt. A new clausal class decidable by hyperresolution. Preprint series, University of Manchester, UK, 2002. Long version of [28].

[28] L. Georgieva, U. Hustadt, and R. A. Schmidt. A new clausal class decidable by hyperresolution. In A. Voronkov, editor, *Proceedings of the 18th International Conference on Automated Deduction (CADE-18)*, volume 2392 of *Lecture Notes in Artificial Intelligence*, pages 260–274. Springer, 2002.

[29] L. Georgieva, U. Hustadt, and R. A. Schmidt. On the relationship between decidable fragments, non-classical logics, and description logics. In I. Horrocks and S. Tessaris, editors, *Proceedings of the 2002 International Workshop on Description Logics (DL'2002)*, pages 25–36. CEUR Workshop Proceedings, Vol. CEUR-WS/Vol-53, 2002.

[30] L. Georgieva, U. Hustadt, and R. A. Schmidt. Hyperresolution for guarded formulae. *Journal of Symbolic Computation*, 2004. To appear.

[31] E. Giunchiglia, F. Giunchiglia, R. Sebastiani, and A. Tacchella. Sat vs. translation based decision procedures for modal logics: A comparative evaluation. *Journal of Applied Non-Classical Logics*, 10(2):145–172, 2000.

[32] E. Grädel, P. Kolaitis, and M. Vardi. On the decision problem for two-variable first-order logic. *Bulletin of Symbolic Logic*, 3:53–69, 1997.

[33] J. Gustafsson. An implementation and optimization of an algorithm for reducing formulas in second-order logic. Technical Report LiTH-MAT-R-96-04, Department of Mathematics, Linköping University, Sweden, 1996.

[34] Hasegawa, H. R., Fujita, and M. Koshimura. Efficient minimal model generation using branching lemmas. In D. McAllester, editor, *Proceedings of the 17th International Conference on Automated Deduction (CADE-17)*, volume 1831 of *Lecture Notes in Artificial Intelligence*, pages 184–199. Springer, 2000.

[35] A. Herzig. *Raisonnement automatique en logique modale et algorithmes d'unification.* PhD thesis, Univ. Paul-Sabatier, Toulouse, 1989.

[36] A. Heuerding, G. Jäger, S. Schwendimann, and M. Seyfried. The Logics Workbench LWB: A snapshot. *Euromath Bulletin*, 2(1):177–186, 1996.

[37] I. Horrocks and P. F. Patel-Schneider. FaCT and DLP. In H. de Swart, editor, *Proceedings of the 7th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'98)*, volume 1397 of *Lecture Notes in Computer Science*, pages 27–30. Springer, 1998.

[38] I. L. Humberstone. Inaccessible worlds. *Notre Dame Journal of Formal Logic*, 24(3):346–352, 1983.

[39] I. L. Humberstone. The modal logic of 'all and only'. *Notre Dame Journal of Formal Logic*, 28(2):177–188, 1987.

[40] U. Hustadt. *Resolution-Based Decision Procedures for Subclasses of First-Order Logic*. PhD thesis, Univ. d. Saarlandes, Saarbrücken, Germany, 1999.

[41] U. Hustadt and R. A. Schmidt. Simplification and backjumping in modal tableau. In H. de Swart, editor, *Proceedings of the 7h International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'98)*, volume 1397 of *Lecture Notes in Artificial Intelligence*, pages 187–201. Springer, 1998.

[42] U. Hustadt and R. A. Schmidt. An empirical analysis of modal theorem provers. *Journal of Applied Non-Classical Logics*, 9(4):479–522, 1999.

[43] U. Hustadt and R. A. Schmidt. Maslov's class K revisited. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction (CADE-16)*, volume 1632 of *Lecture Notes in Artificial Intelligence*, pages 172–186. Springer, 1999.

[44] U. Hustadt and R. A. Schmidt. On the relation of resolution and tableaux proof systems for description logics. In T. Dean, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, volume 1, pages 110–115. Morgan Kaufmann, 1999.

[45] U. Hustadt and R. A. Schmidt. Issues of decidability for description logics in the framework of resolution. In R. Caferra and G. Salzer, editors, *Automated Deduction in Classical and Non-Classical Logics*, volume 1761 of *Lecture Notes in Artificial Intelligence*, pages 191–205. Springer, 2000.

[46] U. Hustadt and R. A. Schmidt. MSPASS: Modal reasoning by translation and first-order resolution. In R. Dyckhoff, editor, *Proceedings of the 9th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2000)*, volume 1847 of *Lecture Notes in Artificial Intelligence*, pages 67–71. Springer, 2000.

[47] U. Hustadt and R. A. Schmidt. Using resolution for testing modal satisfiability and building models. *Journal of Automated Reasoning*, 28(2):205–232, 2002.

[48] U. Hustadt, R. A. Schmidt, and C. Weidenbach. Optimised functional translation and resolution. In H. de Swart, editor, *Proceedings of the 7th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'98)*, volume 1397 of *Lecture Notes in Artificial Intelligence*, pages 36–37. Springer, 1998.

[49] Á. Kurucz, I. Németi, I. Sain, and A. Simon. Undecidable varieties of semilattice-ordered semigroups, of Boolean algebras with operators and logics extending lambek calculus. *Bull. of the IGPL*, 1(1):91–98, 1993.

[50] A. Leitsch. *The Resolution Calculus*. EATCS Texts in Theoretical Computer Science. Springer, 1997.

[51] C. Lutz and U. Sattler. The complexity of reasoning with boolean modal logics. In F. Wolter, H. Wansing, M. de Rijke, and M. Zakharyaschev, editors, *Advances in Modal Logics Volume 3*. CSLI Publications, Stanford, 2002.

[52] C. Lutz, U. Sattler, and S. Tobies. A suggestion of an n-ary description logic. In P. Lambrix, A. Borgida, M. Lenzerini, R. Möller, and P. Patel-Schneider, editors, *Proceedings of the 1999 International Workshop on Description Logics (DL'99)*, pages 81–85. Linköping University, 1999.

[53] F. Massacci. Single step tableaux for modal logics: Computational properties, complexity and methodology. *Journal of Automated Reasoning*, 24(3):319–364, 2000.

[54] W. McCune. The Otter theorem prover, 1995. `http://www.mcs.anl.gov/AR/otter/`.

[55] I. Niemelä. A tableau calculus for minimal model reasoning. In *Proceedings of the 5th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'96)*, volume 1071 of *Lecture Notes in Artificial Intelligence*, pages 278–294. Springer, 1996.

[56] A. Nonnengart. First-order modal logic theorem proving and functional simulation. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI'93)*, volume 1, pages 80–85. Morgan Kaufmann, 1993.

[57] A. Nonnengart, H. J. Ohlbach, and A. Szalas. Quantifier elimination for second-order predicate logic. To appear in *Logic, Language and Reasoning: Essays in honour of Dov Gabbay*, Part I, Kluwer.

[58] A. Nonnengart, G. Rock, and C. Weidenbach. On generating small clause normal forms. In C. Kirchner and H. Kirchner, editors, *Proceedings of the 15th International Conference on Automated Deduction (CADE-15)*, volume 1421 of *Lecture Notes in Artificial Intelligence*, pages 397–411. Springer, 1998.

[59] H. J. Ohlbach. Semantics based translation methods for modal logics. *Journal of Logic and Computation*, 1(5):691–746, 1991.

[60] H. J. Ohlbach. Translation methods for non-classical logics: An overview. *Bulletin of the IGPL*, 1(1):69–89, 1993.

[61] H. J. Ohlbach. Combining Hilbert style and semantic reasoning in a resolution framework. In C. Kirchner and H. Kirchner, editors, *Proceedings of the 15th International Conference on Automated Deduction (CADE-15)*, volume 1421 of *Lecture Notes in Artificial Intelligence*, pages 205–219. Springer, 1998.

[62] H. J. Ohlbach, A. Nonnengart, M. de Rijke, and D. Gabbay. Encoding two-valued nonclassical logics in classical logic. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume II, chapter 21, pages 1403–1486. Elsevier Science, 2001.

[63] H. J. Ohlbach and R. A. Schmidt. Functional translation and second-order frame properties of modal logics. *Journal of Logic and Computation*, 7(5):581–603, 1997.

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

[65] D. A. Plaisted and S. Greenbaum. A structure-preserving clause form translation. *Journal of Symbolic Computation*, 2:293–304, 1986.

[66] A. Riazanov and A. Voronkov. Vampire. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction (CADE-16)*, volume 1632 of *Lecture Notes in Artificial Intelligence*, pages 292–296. Springer, 1999.

[67] A. Riazanov and A. Voronkov. Splitting without backtracking. In B. Nebel, editor, *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pages 611–617. Morgan Kaufmann, 2001.

[68] H. Sahlqvist. Completeness and correspondence in the first and second order semantics for modal logics. In S. Kanger, editor, *Proceedings of the Third Scandinavian Logic Symposium, 1973*, pages 110–143. North-Holland, 1975.

[69] R. A. Schmidt. MSPASS. `http://www.cs.man.ac.uk/~schmidt/mspass/`.

[70] R. A. Schmidt. Decidability by resolution for propositional modal logics. *Journal of Automated Reasoning*, 22(4):379–396, 1999.

[71] R. A. Schmidt and U. Hustadt. A resolution decision procedure for fluted logic. In D. McAllester, editor, *Proceedings of the 17th International Conference on Automated Deduction (CADE-17)*, volume 1831 of *Lecture Notes in Artificial Intelligence*, pages 433–448. Springer, 2000.

[72] S. Schulz. System Abstract: E 0.3. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction (CADE-16)*, volume 1632 of *Lecture Notes in Artificial Intelligence*, pages 297–301. Springer, 1999.

[73] S. Schulz. E: A Brainiac theorem prover. *Journal of AI Communications*, 2002. To appear.

[74] C. Weidenbach. SPASS, 1999. `http://spass.mpi-sb.mpg.de`.

[75] C. Weidenbach. Combining superposition, sorts and splitting. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume II, chapter 27, pages 1965–2013. Elsevier, 2001.

[76] N. K. Zamov. Modal resolutions. *Soviet Mathematics*, 33(9):22–29, 1989. Translated from *Izv. Vyssh. Uchebn. Zaved. Mat.* **9** (328) (1989) 22–29.