Resolution-Based Methods for Modal Logics

HANS DE NIVELLE, *Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany.*
E-mail: nivelle@mpi-sb.mpg.de

RENATE A. SCHMIDT, *Department of Computer Science, University of Manchester, Manchester M13 9PL, United Kingdom.*
E-mail: schmidt@cs.man.ac.uk

ULLRICH HUSTADT, *Centre for Agent Research and Development, Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom.*
E-mail: U.Hustadt@doc.mmu.ac.uk

Abstract

In this paper we give an overview of resolution methods for extended propositional modal logics. We adopt the standard translation approach and consider different resolution refinements which provide decision procedures for the resulting clause sets. Our procedures are based on ordered resolution and selection-based resolution. The logics that we cover are multi-modal logics defined over relations closed under intersection, union, converse and possibly complementation.

Keywords: modal logic, solvable classes, guarded fragment, description logic, inference methods, resolution, tableaux, model generation

1 Introduction

Modal logics are very popular and appear in various disguises in many areas of computer science, including knowledge representation, the field of logics of programs, computational linguistics and agent based systems. While decidability is an important criterion in many of these areas increasingly more expressive modal logics which allow complex relational parameters of modal operators are being used. Consider an example from knowledge representation and linguistics domains. Here the universes of frames contain arbitrary elements instead of worlds. If E denotes the eats relation and C is the set of cheeses, then $\langle E\rangle C$ can be interpreted as denoting the set of cheese eaters. An expression which requires complex relational parameters is the set of cheese lovers: $[\neg(E \wedge L)]\neg C$, where L denotes the likes relation. We have $x \in [\neg(E \wedge L)]\neg C$ iff for any $y \in C$, both $E(x,y)$ and $L(x,y)$ are true. In words, cheese lovers are people who eat and like every cheese. The meaning of $x \in [E \wedge L]C$ would be 'everything that x eats and likes is cheese'. These kinds of expressions can be formulated in the logics we consider in this paper.

We focus on subsystems of the multi-modal logic $K_{(m)}(\cap, \cup, ^{-}, ^{\smile})$ which is defined over families of relations closed under intersection, union, complementation and converse. $K_{(m)}(\cap, \cup, ^-, ^\smile)$ extends Boolean modal logic [17] with converse on relations. It encompasses very many standard modal logics such as *K*, *KT*, *KD*, *KB*, *KTB*, and *KDB*, their independent joins, as well as the basic tense logic K_t and logics of philosophical interest, such as logics expressing inaccessibility, sufficiency, or both necessity and sufficiency, see e.g. [18, 23, 24]. Certain forms of interactions, for example, inclusions among relations, are covered as well. $K_{(m)}(\cap, \cup, ^-, ^\smile)$ is related to the description logic $\mathcal{ALB}$ which was first described in [28] and contains a large class of well known description logics.

We concentrate on translation-based resolution methods for modal logics. This means that we take a modal formula, translate it into classical logic through the Kripke-semantics, and then apply some variant of resolution to it. Translation-based approaches are sometimes regarded as being inferior to tableaux-based approaches, or other special-purpose inference approaches. Arguably recent advances in the implementation of tableaux-based modal theorem provers make it harder to motivate the endeavour of translation into first-order logic. Another criticism often brought forward is the difficulty of reading resolution proofs (this is not true in general, see [28]). From our perspective the combination of translation and first-order resolution has a number of advantages, as this paper aims to show. Some obvious advantages of translation approaches are the following. Any modal logic which can be embedded into first-order logic can be treated. The translations are straightforward, and can be obtained 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 (for example, Bliksem [10], SPASS [40], and Otter [34]). The translation approach is generic, it can handle first-order modal logics, undecidable modal logics, for example, de Rijke's dynamic modal logic [11], and combinations of modal and non-modal logics. In all cases we can at least ensure soundness and completeness. For a large class of expressive modal and description logics, resolution provers provide decision procedures, and often the same refinements decide also first-order generalisations such as the guarded fragment or Maslov's class K [14, 26].

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 will 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. Surveys of the different translation methods are Ohlbach [35, 36] and Ohlbach, Nonnengart and Gabbay [37].

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. [12, 13], Leitsch [30], Hustadt and Schmidt [28, 29]). We will describe three resolution decision procedures: an ordered resolution decision procedure for a class of clauses induced by $K_{(m)}(\cap, \cup, ^-, ^\smile)$ (Section 5), an ordering refinement combined with a selection function for the guarded fragment (Section 6), and a refinement which relies solely on the selection of negative literals for certain extensions of $K_{(m)}(\cap, \cup, \smile)$ (Section 7). The latter refinement has the property that for many modal logics its derivations resemble those of tableaux calculi. As with tableaux-based procedures our selection-based procedure can be used for the automatic construction of finite models for satisfiable input formulae. In Section 8 we define a semantic tableaux calculus for the logic $K_{(m)}(\cap, \cup, \smile)$ which is derived from the selection-based resolution procedure. We also consider the relationship to single step prefixed tableaux calculi and prove a number of simulation results. Preliminary definitions are given in Sections 2, 3 and 4. Section 2 contains definitions of the notational conventions and basic concepts. Of particular importance is the structural transformation of formulae. Section 3 defines the syntax and semantics of the logic $K_{(m)}(\cap, \cup, ^-, \smile)$ and specifies the standard translation mapping into first-order logic. A general framework of ordered resolution and selection is described in Section 4.

This overview is based on the papers [14, 28, 29]. Some results have been improved and others are new. The definition of the class DL* in Section 5, generalises the class of DL-clauses from [28]. Section 7 includes a new complexity result. The results for extensions of $K_{(m)}(\cap, \cup, \smile)$ with frame properties are slightly more general than in [29]. The close correspondence between selection-based resolution (or hyperresolution) and special purpose tableaux calculi is also mentioned in [13, 28, 29]. A novelty are the tableaux calculi which we have been able to extract from the selection-based resolution procedure. These are related to calculi for the corresponding description logics [22, 21], but they do not compile relational formulae away.

2 Preliminary Definitions and Conventions

Throughout, our notational convention is the following: x, y, z are the letters reserved for first-order variables, s, t, u, v for terms, a, b for constants, f, g, h for function symbols, 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 we use the letter N. The Greek letters φ, ψ, ϕ are reserved for modal or first-order formulae, and α, β, γ are reserved for relational formulae.

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 (resp. two) then we call this literal a *unary literal* (resp. *binary literal*). A clause with one literal is a *unit clause* (or unit). If this literal is a unary (resp. binary) literal then the clause will be called a unary (resp. binary) unit clause. In this paper *clauses* are assumed to be sets of literals. The empty clause will be denoted by $\emptyset$. The components in the variable partition of a clause are called *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.

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 will be denoted by $\mathrm{Pos}(\varphi)$.

The *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 $Q_\lambda(x_1, \ldots, x_n)$, where $x_1, \ldots, x_n$ 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.[1] Let

$$\begin{aligned} \mathrm{Def}^+_\lambda(\varphi) &= \forall x_1 \ldots x_n \ (Q_\lambda(x_1, \ldots, x_n) \rightarrow \varphi|_\lambda) \quad \text{and} \\ \mathrm{Def}^-_\lambda(\varphi) &= \forall x_1 \ldots x_n \ (\varphi|_\lambda \rightarrow Q_\lambda(x_1, \ldots, x_n)). \end{aligned}$$

The *definition* of Q_λ is the formula

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

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

$$\begin{aligned} \mathrm{Def}_\emptyset(\varphi) &= \varphi \quad \text{and} \\ \mathrm{Def}_{\Lambda \cup \{\lambda\}}(\varphi) &= \mathrm{Def}_\Lambda(\varphi[Q_\lambda(x_1, \ldots, x_n) \mapsto \lambda]) \wedge \mathrm{Def}_\lambda(\varphi), \end{aligned}$$

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 2.1 (e.g. Plaisted and Greenbaum [39])
Let φ be a first-order formula.

1. φ is satisfiable iff $\mathrm{Def}_\Lambda(\varphi)$ is satisfiable, for any $\Lambda \subseteq \mathrm{Pos}(\varphi)$.
2. $\mathrm{Def}_\Lambda(\varphi)$ can be computed in polynomial time.

3 The Modal Logic $\boldsymbol{K}_{(m)}(\cap, \cup, ^-, ^\smile)$

$K_{(m)}(\cap, \cup, ^-, ^\smile)$ is the multi-modal logic defined over families of binary relations closed under intersection, union, complementation and converse.

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,

[1] 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.

for example, $\varphi \to \psi = \neg\varphi \vee \psi$ or the universal modality is $[*] = [r_j \vee \neg r_j]$, for some relational variable r_j.

We will also consider logics with fewer relational operations. Formally, by a logic in-between K and $K_{(m)}(\cap, \cup, ^-, ^\smile)$ we mean a logic $K_{(m)}(\star_1, \ldots, \star_k)$ where $m \geq 1$, $1 \leq k \leq 4$ and the $\star_i$ are distinct operations from $\{\cap, \cup, ^-, ^\smile\}$.

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:

$$
\begin{aligned}
R(\alpha \wedge \beta) &= R(\alpha) \cap R(\beta) & \qquad R(\neg\alpha) &= \overline{R(\alpha)} \\
R(\alpha \vee \beta) &= R(\alpha) \cup R(\beta) & \qquad R(\alpha^\smile) &= R(\alpha)^\smile.
\end{aligned}
$$

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

$$
\begin{aligned}
\iota(\bot) &= \emptyset & \iota(\top) &= W \qquad \iota(\neg\varphi) = \overline{\iota(\varphi)} \\
\iota(\varphi \wedge \psi) &= \iota(\varphi) \cap \iota(\psi) & \iota(\langle\alpha\rangle\varphi) &= \{x \mid \exists y \in W\ (x, y) \in R(\alpha) \wedge y \in \iota(\varphi)\} \\
\iota(\varphi \vee \psi) &= \iota(\varphi) \cup \iota(\psi) & \iota([\alpha]\varphi) &= \{x \mid \forall y \in W\ (x, y) \in R(\alpha) \to y \in \iota(\varphi)\}.
\end{aligned}
$$

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

The standard translation of $K_{(m)}(\cap, \cup, ^-, ^\smile)$ into first-order logic follows the semantic definition and is therefore given by the following.

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

Relational formulae are translated according to:

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

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 3.1
Let L be a logic in-between K and $K_{(m)}(\cap, \cup, ^-, ^\smile)$. For any modal formula φ, φ is satisfiable in L iff $\Pi(\varphi)$ is first-order satisfiable.

In order to keep the presentation simple, modal formulae are assumed to be in negation normal form. This means that in every subformula of the form $\neg\varphi$, φ is a propositional variable. The negation normal form of any modal formula is obtained as usual, namely, by moving negation symbols inwards as far as possible (using De Morgan's laws, $\neg\langle\alpha\rangle\psi \leftrightarrow [\alpha]\neg\psi$ and $\neg[\alpha]\psi \leftrightarrow \langle\alpha\rangle\neg\psi$, and $\neg(\alpha^{\smile}) \leftrightarrow (\neg\alpha)^{\smile}$) and eliminating double negations.

4 The Resolution Framework

In this paper we will make use of A-ordered resolution, extended with selection. A-ordered resolution is well-known and widely used in resolution decision procedures [12, 13, 5, 34, 30, 26]. It follows from the results in Bachmair and Ganzinger [3, 4] that A-ordered resolution can be combined with a selection function. This selection function can override the A-ordering, give preference to inferences with negative literals. A-ordered resolution with selection is controlled by two parameters: an A-ordering and a selection function. An A-ordering is an ordering $\succ$ on atoms, which satisfies the following condition: For all atoms A, B and for all substitutions σ, $A \succ B$ implies $A\sigma \succ B\sigma$. For a literal $L = (\neg)A$ let $\mathrm{at}(L) = A$. A-orderings are extended to literals by $L \succ L'$ iff $\mathrm{at}(L) \succ \mathrm{at}(L')$. If one uses orderings that do not ignore the negation sign (these are called L-orderings), one does not loose completeness [7]. However L-orderings cannot be combined with selection. Given an A-ordering $\succ$, we define the *maximal literals* in a clause in the standard way: A literal L in a clause C is maximal in C, if there is no literal L' in C, for which $L' \succ L$.

Let $\succ$ be an A-ordering. A *selection function* S, based on $\succ$, is a function which assigns to each clause C a non-empty set of its literals, such that one of the following holds:

(4.1) Either $S(C)$ contains a negative literal, or

(4.2) $S(C)$ contains all the $\succ$-maximal literals of C.

No further restrictions are imposed on the selection function. If the selection function always prefers the second alternative, one has just A-ordered resolution. If the selection function always selects only the negative literals in non-positive clauses, then the restriction simulates A-ordered hyperresolution. Based on a selection function S, resolution and factoring can be defined as follows:

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, and (ii) $A_1 \in S(C \vee A_1)$ and $\neg A_2 \in S(\neg A_2 \vee D)$. Then the clause $(C \vee D)\sigma$ is a *resolvent*.

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) $A_1 \in S(C \vee A_1 \vee A_2)$. Then the clause $(C \vee A_1)\sigma$ is called a *factor* of $C \vee A_1 \vee A_2$.

The combination of selection-based resolution and factoring forms a complete refutation system for clause sets.

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*.

Simplification and Splitting

In the previous section we explained where the clauses come from. In this section we explain how to get rid of them. In order to obtain termination, one needs redundancy criteria. Let C and D be clauses. Clause C *subsumes* D if $|C| \leq |D|$, and there exists a substitution σ, such that $C\sigma \subseteq D$. Without the length-restriction factors would be subsumed by their parents. This would result in deletion of all factors. Since the factoring rule is necessary to completeness, deleting all factors would result in incompleteness. Determining whether or not clause C subsumes clause D, is NP-complete. A *condensation* of C is a minimal subset D of C, such that D subsumes C. One can show that condensations are unique up to renaming. Determining whether or not a clause is condensed, is NP-complete. Computing the condensation is NP-hard. In practice, NP-hardness does not cause problems, since the clauses are short ($< \log\log$) in comparison to the number of clauses. A clause C is a *tautology* if it contains a complementary pair of literals A and $\neg A$.

Let N be a clause set. A *saturation* of N is a clause set N_∞, such that, for every non-tautological clause C in N, there is a clause D in N_∞, such that D subsumes C, and for each non-tautological clause C, that is derivable from clauses in N_∞, there is a clause D in N_∞, such that D subsumes C.

For selection based resolution the following holds.

THEOREM 4.1
For every clause set N, and every saturation N_∞ of N the following holds: N is unsatisfiable iff N_∞ contains the empty clause.

This follows from the results in Bachmair and Ganzinger [3, 4]. This completeness allows us to freely delete tautologies and subsumed clauses, or replace clauses by condensations. In general it is possible to use stronger notions of redundancy. One can define a clause to be redundant if it is implied by a finite set of strictly smaller clauses (under an appropriate extension of $\succ$ to clauses), see [3, 4].

Our notion of saturation is not appropriate for building into a real theorem prover, because it does not model the time aspect. A clause may become redundant only after some time, after it has been used for deriving clauses that occur in the proof.

The *splitting rule* is a rule that is borrowed from semantic tableaux. Let N be a set of clauses containing a clause C, that has two split components C_1 and C_2. Then, instead of trying to refute N one tries to refute $N \cup \{C_1\}$ and $N \cup \{C_2\}$ (or $N \cup \{C_1\}$ and $N \cup \{C_2, \neg C_1\}$, if C_1 is a ground clause). Note that in both sets, the original clause C has become redundant. The splitting rule can be essentially simulated in the resolution context by introducing a new propositional symbol. If $C_1 \vee C_2$ is a clause that can be split into two split components C_1 and C_2, then it is possible to replace $C_1 \vee C_2$ by two clauses $C_1 \vee q$, and $\neg q \vee C_2$. q is made minimal in the A-ordering, and $\neg q$ is selected. In most cases this is easier to implement than the full splitting rule.

5 Ordered Resolution for $\boldsymbol{K}_{(m)}(\cap, \cup, ^{-}, ^{\smile})$

Many modal logics naturally translate into decidable fragments of first-order logic. For example the basic logic K translates into the two-variable fragment, and into the guarded fragment. By constructing decision procedures for these decidable fragments, one obtains generic decision procedures for modal logics. We consider two classes. One is a clause fragment based on the two-variable fragment, called DL*. This fragment is a variation of the class of DL-clauses, that was introduced in Hustadt and Schmidt [28] with the purpose of handling expressive description logics. The other one is the guarded fragment, which was introduced by Andréka, Van Benthem and Németi [2] as the 'modal subset of first-order logic'. Although it did not quite meet the ambitious goals, it is an important fragment, containing many modal logics.

The class of DL*-clauses is related to the class $\mathcal{S}^+$ in Fermüller et al. [12]. This class was introduced there as the clause fragment belonging to the two-variable fragment. The class $\mathcal{S}^+$ can only be decided by a non-liftable ordering [8], or by an A-ordering combined with a rule called monadisation [12]. Since we try to root our approach on the common basis of liftable orderings, we slightly restrict the class, so that it can be decided by a liftable ordering. The restriction is still general enough to contain the clause translations of the Π-transformation of the modal formulae in $K_{(m)}(\cap, \cup, ^{-}, ^{\smile})$.

We now introduce the clause fragment DL*. In order to simplify the exposition, we assume that all clauses are maximally split. The notions can be easily adopted for clauses with more than one split component.

Let C be a clause. It is a *DL*-clause* if

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, and
4. every binary literal (even if it has no functional terms) contains all variables of C.

Observe that 3. implies that if C contains a functional ground term, then C is ground. The difference with $\mathcal{S}^+$ is Condition 4. For $\mathcal{S}^+$, Condition 4 would be (4a): Every clause C has a literal containing all variables of C. Condition 4 forbids the following problematic clauses, which are allowed by Condition 4a: $P(x,x) \vee Q(x,y)$ and $\neg P(x,x) \vee R(x,y)$. In order to stay within $\mathcal{S}^+$, one would have to block the inference based on $P(x,x)$ and $\neg P(x,x)$, since this would result in the clause $Q(x,y) \vee R(x,z)$, which contains more variables than each of the parent clauses. However no A-ordering can put $Q(x,y) \succ P(x,x)$, for all predicate symbols P and Q.

Examples of DL*-clauses include ground clauses, and

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

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 in principle belong to DL*, but is not maximally split.

THEOREM 5.1
Over a finite signature[2] there are only finitely many maximally split DL*-clauses (modulo variable renaming).

The proof is similar to the proof for the class of DL-clauses in Hustadt and Schmidt [28]. 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.

THEOREM 5.2
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.

PROOF. Let a be maximal arity of any function symbol. Because any clause contains at most a variables, the number of possible terms is bounded by $(s+a)+(s+a)^{a+1} \leq (s+a)^{a+2}$. The number of possible atoms is then equal to $s((s+a)^{a+2})^2 \leq (s+a)^{2a+5}$. The number of possible literals equals $2(s+a)^{2a+5} \leq (s+a)^{2a+6}$. Consequently, the number of non-equivalent clauses is bounded by $2^{(s+a)^{2a+6}} = 2^{2^{(2a+6)\log(s+a)}}$. ∎

The reduction of modal formulae to sets of DL*-clauses makes use of a structural transformation introducing new names for subformulae corresponding to non-literal subformulae of the original modal formula. For a given modal formula φ and its translation into first-order logic $\varphi' = \Pi(\varphi)$, we apply the mapping Def_Λ with

$$\Lambda = \{\lambda \mid \text{there is a non-literal subformula } \varphi|_{\lambda'} \text{ of } \varphi \text{ and } \varphi'|_\lambda = \Pi(\varphi|_{\lambda'})\}.$$

For example, the definition corresponding to a subformula $\langle r_j\rangle p$ is

$$\forall x\,(Q_{\langle r_j\rangle p}(x) \rightarrow \exists y\ (R_j(x,y) \wedge P(y))).$$

The formula

$$\exists x \forall y\,((\neg R_1(x,y) \wedge R_2(x,y)) \rightarrow \exists z\,(\neg R_1(y,z) \wedge R_2(y,z) \wedge P(z))), \qquad (*)$$

which is a translation of the modal formula $[\neg r_1 \wedge r_2]\langle\neg r_1 \wedge r_2\rangle p$ results in the following set of definitions, together with $\exists x\, Q_{[\alpha]\langle\alpha\rangle p}(x)$.

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

Here α is used as an abbreviation for $\neg r_1 \wedge r_2$. Notice that one new symbol Q_α was used for the positive and negative occurrences of the subformula $\neg R_1(x,y) \wedge R_2(x,y)$.

THEOREM 5.3
Let φ' be a first-order formula that results from the 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.

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

PROOF. Not difficult. ∎

In order to decide the class DL*, we use the following A-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 $>_d$ ordering originates from Fermüller et al. [12]. The selection function S is completely determined by $\succ$, so there is no preferred selection of negative literals.

We now give the clausal normal form of the formula (*) above. The maximal literals are marked with *. These are the literals that can potentially be resolved or factored upon.

$$
\begin{array}{l}
 Q_{[\alpha]\langle\alpha\rangle p}(a)^* \\
\neg Q_{[\alpha]\langle\alpha\rangle p}(x) \vee \neg Q_\alpha(x, y)^* \vee Q_{\langle\alpha\rangle p}(y) \\
\neg Q_{\langle\alpha\rangle p}(x) \vee Q_\alpha(x, f(x))^* \\
\neg Q_{\langle\alpha\rangle p}(x) \vee P(f(x))^* \\
\neg Q_\alpha(x, y)^* \vee \neg R_1(x, y)^* \\
\neg Q_\alpha(x, y)^* \vee R_2(x, y)^* \\
 R_1(x, y)^* \vee \neg R_2(x, y)^* \vee Q_\alpha(x, y)^*
\end{array}
$$

In the last three clauses there is more than one maximal literal. This could be prevented by completing $\succ$ with an ordering on atoms. In that case it is necessary to distinguish equivalent from incomparable literals. Instead of $\succ$, one would have to define $\succeq$. Then $A \succ B$ would have to be defined as $A \succeq B$ and $A \not\preceq B$. In the case that $A \succeq B$ and $A \preceq B$, one can try to use a second ordering for establishing a priority.

In order to prove that the procedure that we described is indeed a decision procedure we have to show that it is complete, and terminating. The completeness follows from Theorem 4.1. Termination is a consequence of Theorem 5.1, and the fact that the restriction derives only clauses that are within DL*, or that can be split. This fact is obtained by a case analysis, similar as in [28]. Therefore:

THEOREM 5.4
Let L be a logic in-between K and $K_{(m)}(\cap, \cup, {}^-, {}^\smile)$. Let N be the clausal form of $\mathrm{Def}_\Lambda \Pi(\varphi)$, where φ is any modal formula in L. Then:

1. Any derivation from N terminates in double exponential time.
2. φ is unsatisfiable in L iff the saturation of N contains the empty clause.

This result covers actually a larger class of modal logics. Boolean modal logic, and hence also $K_{(m)}(\cap, \cup, {}^-, {}^\smile)$, is expressive enough to allow for frame properties to be specified by relational formulae. Implication of relational formulae can be defined by $(\alpha \to \beta) = [\alpha \wedge \neg\beta]\bot$ [38]. Hence, the symmetry of the accessibility relation R_1 associated with r_1 can be specified by $r_1 \to r_1^\smile$.

If Δ is a set of relational frame properties then $L\Delta$ will denote the logic characterised by the class of frames satisfying the conjunction of properties in Δ.

COROLLARY 5.5
Let L be a logic in-between K and $K_{(m)}(\cap, \cup, {}^-, {}^\smile)$. Let Δ be the Boolean combination of relational inclusions or equivalences expressed over intersection, union, complementation and converse. Suppose φ is any modal formula and N is the clausal form of $\mathrm{Def}_\Lambda \Pi(\varphi)$. Then:

1. Any derivation from $N \cup \Delta$ terminates in double exponential time.
2. φ is unsatisfiable in $L\Delta$ iff the saturation of $N \cup \Delta$ contains the empty clause.

The decidability result for the classes DL* and DL allows for a slightly more general result, which includes reflexivity and irreflexivity. Modal and relational formulae with positive occurrences of relational composition can also be embedded into the class DL*. Moreover, relational properties such as $\forall xy\,(R_1(x,y) \to R_2(x,x))$ are covered by the class $\mathcal{S}^+$.

6 Ordered Resolution for the Guarded Fragment

In this section we use ordered resolution with selection as a decision procedure for the guarded fragment. The guarded fragment was first shown decidable by Andréka, Németi and Van Benthem [1]. Grädel [20] has shown that the satisfiability problem for the guarded fragment is DEXPTIME-complete. There it was also shown that the guard condition is necessary only for the universal quantifiers, when the formula is in negation normal form. A resolution decision procedure for the guarded fragment was first established in de Nivelle [9]. In Ganzinger and de Nivelle [14] the method was adapted to the guarded fragment with equality. It is shown there that the complexity of the resolution decision procedure is consistent with the complexity given in [20]. The decision procedure that we give here is based on the one in [14].

A first-order formula is in the *guarded fragment* if it is function free, and every quantification has form $\forall \overline{x}\,(G \to \psi)$, or $\exists \overline{x}\,(G \wedge \psi)$. Here G is an atom containing all free variables of ψ, and $\overline{x}$ is a sequence of variables.

We use the following clausal normal form. A clause C is a *guarded clause* if

1. there is no nesting of function symbols,
2. every functional term in C contains all variables of C, and
3. if C contains variables, then there is a negative, function-free literal that contains all variables of C. Such a literal is called a *guard* literal.

As is the case with the class of DL*-clauses, there is only a finitely bounded set of guarded clauses.

THEOREM 6.1 (Ganzinger and de Nivelle [14])
Over a finite signature the number of possible guarded clauses is of order 2^{2^s}, where s is the size of the signature.

For the reduction to clausal normal form we assume that a guarded formula φ is in negation normal form. The reduction of φ into guarded clauses uses a structural transformation Def_Λ with

$$\Lambda = \{\lambda \mid \lambda \text{ is a position in } \varphi \text{ of a formula of the form } \forall \overline{x}\,(G \to \psi)\}.$$

It can be shown that this structural transformation preserves the guarded fragment. The definitional formula that defines a guarded formula $\forall\overline{x}\,(G \rightarrow \psi)$, has the form

$$\forall\overline{y}\,(Q_{\forall\overline{x}(G\rightarrow\psi)}(\overline{y}) \rightarrow \forall\overline{x}\,(G \rightarrow \psi)).$$

Every variable in $\overline{y}$ and $\overline{x}$ occurs in G. This formula is not guarded by itself but it is equivalent to the following formula, which is guarded: $\forall\overline{xy}\,(G \rightarrow (Q_{\forall\overline{x}(G\rightarrow\psi)}(\overline{y}) \rightarrow \psi))$.

Formulae in $K_{(m)}(\cap,\cup,\smile)$ are translated by Π into the guarded fragment. Negations of accessibility relations would be problematic. For example, $[\neg r]p$ is translated into $\exists x\forall y\,(\neg R(x,y) \rightarrow P(y))$. This formula is not guarded. The formula $[(r_1 \wedge r_2) \vee r_3^{\smile}]p$ is translated into $\exists x\forall y\,((R_1(x,y) \wedge R_2(x,y)\ ) \vee R_3(y,x) \rightarrow P(y))$. This formula is not guarded either, however, it is equivalent to the guarded formula:

$$\exists x\forall y\,(R_1(x,y) \rightarrow (R_2(x,y) \rightarrow P(y))) \wedge \forall y\,(R_3(y,x) \rightarrow P(y)).$$

We show that this is in general the case for formulae in $K_{(m)}(\cap,\cup,\smile)$. The mapping Π translates formulae of $K_{(m)}(\cap,\cup,\smile)$ into first-order formulae in which the quantifications have the form $\forall\overline{x}\,(G \rightarrow \psi)$. In this, G is a relational expression without negation and function symbols, in which each atom contains all free variables of ψ. This G can be translated into disjunctive normal form,

$$(G_{1,1} \wedge \ldots \wedge G_{1,l_1}) \vee \ldots \vee (G_{n,1} \wedge \ldots \wedge G_{n,l_n}).$$

The $G_{i,j}$ are atoms, containing all free variables of ψ. Then $\forall\overline{x}\,(G \rightarrow \psi)$ is equivalent to

$$\forall\overline{x}\,((G_{1,1} \wedge \ldots \wedge G_{1,l_1}) \rightarrow \psi) \wedge \ldots \wedge \forall\overline{x}\,((G_{n,1} \wedge \ldots \wedge G_{n,l_n}) \rightarrow \psi),$$

which is in turn equivalent to

$$\forall\overline{x}\,(G_{1,1} \rightarrow (\ldots \rightarrow (G_{1,l_1} \rightarrow \psi))) \wedge \ldots \wedge \forall\overline{x}\,(G_{n,1} \rightarrow (\ldots \rightarrow (G_{n,l_n} \rightarrow \psi))).$$

The $G_{i,1}$ are well-formed guards.

In order to obtain a decision procedure for the guarded fragment, we make use of the ordering $\succ$ of the previous section, combined with selection of negative literals.

1. If C is a non-ground clause without functional terms, then $S(C)$ contains all guards of C.
2. If C is a clause with functional terms, then $S(C)$ contains all literals with functional terms.

It is easily checked that this is a valid selection function for guarded clauses. If C is a non-ground clause, then it has at least one guard. Because this guard is negative, it is possible to select it. If C contains functional terms, then some of the literals containing functional terms are $\succ$-maximal. Because of this it is possible to select these literals.

The formula $\exists x\exists y\,(R_1(x,y) \wedge R_2(x,y) \wedge \forall z\,(R_1(y,z) \rightarrow R_2(y,z) \rightarrow P(z)))$, which is a translation of $\langle r_1 \wedge r_2\rangle[r_1 \wedge r_2]p$ results in the following formula.

$$\begin{aligned}&\exists x\exists y\,(R_1(x,y) \wedge R_2(x,y) \wedge Q_{[r_1\wedge r_2]p}(y))\\ &\quad \wedge \forall yz\,((R_1(y,z) \wedge R_2(y,z)) \rightarrow (Q_{[r_1\wedge r_2]p}(y) \rightarrow P(z))).\end{aligned}$$

The clausal normal form consists of the clauses

$$
\begin{aligned}
&R_1(a,b)^* \\
&R_2(a,b)^* \\
&Q_{[r_1 \wedge r_2]p}(b)^* \\
\neg &R_1(x,y)^* \vee \neg R_2(x,y)^* \vee \neg Q_{[r_1 \wedge r_2]p}(x) \vee P(y).
\end{aligned}
$$

The literals marked with * are the maximal literals. The restriction could be completed, as in the previous section, in order to obtain a more total ordering. The termination proof is analogous to the proof for DL-clauses. The main difficulty is to prove that the restriction preserves the guarded fragment. For this we refer to Ganzinger and de Nivelle [14]. Consequently:

THEOREM 6.2
Let L be a logic in-between K and $K_{(m)}(\cap, \cup, \smile)$. Let N be the clausal form of $\mathrm{Def}_\Lambda \Pi(\varphi)$, where φ is any modal formula in L. Then:

1. Any derivation from N terminates in double exponential time.
2. φ is unsatisfiable in L iff the saturation of N contains the empty clause.

7 Selection-Based Resolution for $\boldsymbol{K}_{(m)}(\cap, \cup, \smile)$

$K_{(m)}(\cap, \cup, \smile)$ 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 [29].

Here new names are introduced for all non-atomic subformulae of the translation of a modal formula, that is, we use Def_Λ where Λ is the subset of positions in φ' (the first-order translation) which correspond to non-atomic subformulae of φ (the original modal formula). Moreover Def_Λ introduces the same symbol for variant subformulae with the same polarity. Because, by assumption, φ is in negation normal form, all occurrences of non-atomic subformulae of φ' with one free variable have positive polarity. This means $\mathrm{Def}_\lambda(\varphi') = \mathrm{Def}^+_\lambda(\varphi')$ 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 will introduce for the subformulae of $[\alpha]\langle\alpha\rangle p$ with $\alpha = r_1 \wedge r_2$ the definitions (in addition to $\exists x\, Q_{[\alpha]\langle\alpha\rangle p}(x)$):

$$
\begin{aligned}
&\forall x\, (Q_{[\alpha]\langle\alpha\rangle p}(x) \rightarrow \forall y\, (Q^n_\alpha(x,y) \rightarrow Q_{\langle\alpha\rangle p}(y))) \\
&\forall x\, (Q_{\langle\alpha\rangle p}(x) \rightarrow \exists y\, (Q^p_\alpha(x,y) \wedge P(y))) \\
&\forall xy\, (Q^p_\alpha(x,y) \rightarrow (R_1(x,y) \wedge R_2(x,y))) \\
&\forall xy\, ((R_1(x,y) \wedge R_2(x,y)) \rightarrow Q^n_\alpha(x,y)).
\end{aligned}
\tag{7.1}
$$

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

In order to characterise the induced class of clauses we introduce some more notation. We denote introduced predicate symbols by Q_ψ and Q^p_α or Q^n_α, where Q_ψ represents an occurrence of a modal subformula ψ and $Q^{p/n}_\alpha$ represents a positive/negative occurrence of a relational subformula α. We also find it convenient to use the notation

$$\begin{array}{rl} \mathcal{P}(s) & \text{for some literal in } \{P_i(s), Q_\psi(s)\}_{i,\psi}, \text{ and} \\ \mathcal{R}(s,t) & \text{for some literal in } \{R_j(s,t), R_j(t,s), Q^{p/n}_\alpha(s,t), Q^{p/n}_\alpha(t,s)\}_{j,\alpha}. \end{array}$$

Note the order of the arguments in $\mathcal{R}(s,t)$ is not fixed. 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

$$\neg Q_\psi(x) \vee \neg R_j(y,x) \vee Q_\chi(y) \quad \text{and} \quad \neg Q_\psi(x) \vee \neg Q^n_\alpha(x,y) \vee Q_\chi(y)$$

are instances of $\neg Q_\psi(x) \vee \neg\mathcal{R}(x,y) \vee \mathcal{P}(y)$.

Thus, all input clauses have one of the following forms.

$$\begin{array}{ll} \mathcal{P}(a) & \\ \neg Q_\psi(x)^* \vee \neg P_i(x)^* & \text{if } \psi = \neg p_i \\ \neg Q_\psi(x)^* \vee \mathcal{P}(x)\ [\vee\ \mathcal{P}(x)] & \text{if } \psi = \phi_1 \wedge[\vee]\ \phi_2 \\ \neg Q_\psi(x)^* \vee \neg\mathcal{R}(x,y)^*\ [\vee\ \mathcal{P}(y)] & \text{if } \psi = [\alpha]\phi\ [\psi = [\alpha]\bot] \\ \neg Q_\psi(x)^* \vee \mathcal{P}(f(x)) & \\ \neg Q_\psi(x)^* \vee \mathcal{R}(x, f(x)) & \text{if } \psi = \langle\alpha\rangle\phi \\ \neg Q^p_\alpha(x,y)^* \vee \mathcal{R}(x,y)\ [\vee\ \mathcal{R}(x,y)] & \text{if } \alpha = \beta_1 \wedge[\vee]\ \beta_2 \text{ has pos. polarity} \\ Q^n_\alpha(x,y) \vee \neg\mathcal{R}(x,y)^*\ [\vee\ \neg\mathcal{R}(x,y)^*] & \text{if } \alpha = \beta_1 \wedge[\vee]\ \beta_2 \text{ has neg. polarity.} \end{array} \tag{7.2}$$

The literals marked with * are the selected literals.

The minimal calculus which we will use 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. The resolution rule is the following:

Resolution with maximal selection:

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

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*.

The inference rules of our calculus, denoted by $R^{\mathcal{MOD}}$, are the above resolution rule, positive factoring, splitting and at least tautology deletion. All derivations in $R^{\mathcal{MOD}}$ 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 (a concrete example is given in Figure 2 below) contain a selected literal no factoring steps are possible and 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
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.

PROOF. Every resolution inference step with a definitional clause from the input set and ground unit clauses of the form $\mathcal{P}(s)$ or $\mathcal{R}(s, f(s))$ yields a ground clause which can be split into ground unit clauses of the required form. ∎

In general, s will be a nested non-constant functional ground term, which is usually undesirable, because in most situations this causes unbounded computations. However, as the next theorem proves, 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.

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

THEOREM 7.2
Let φ be any $K_{(m)}(\cap, \cup, \smile)$-formula and let N be the clausal form of $\mathrm{Def}_\Lambda\Pi(\varphi)$. Then:

1. Any $R^{\mathcal{MOD}}$-derivation from N terminates.
2. φ is unsatisfiable in $K_{(m)}(\cap, \cup, \smile)$ iff the $R^{\mathcal{MOD}}$-saturation of N contains the empty clause.

PROOF. 2. follows from the soundness and refutational completeness of ordered resolution with selection (Theorem 4.1).

For 1., define a dependency relation $\succ_d$ on the predicate symbols by $S_1 \succ_d S_2$, if there is a definition $\psi \to \phi$ in $\mathrm{Def}_\Lambda\Pi(\varphi)$ such that S_1 occurs in ψ and S_2 occurs in ϕ. An additional restriction is that if Q_ψ is the symbol introduced for a diamond formula ψ, and Q_ϕ is the symbol introduced for a box formula ϕ, and ψ and ϕ occur at the same modal depth in φ, then $Q_\psi \succ_d Q_\phi$. Moreover, let `tt` be a new symbol smaller than all predicate symbols. (For example, for (7.1) the dependency relation is depicted in Figure 1. That is, $Q_{[\alpha]\langle\alpha\rangle p} \succ_d Q_{\langle\alpha\rangle p}$, and so on.) Let $\succ_D$ be any ordering on the predicate symbols in $\mathrm{Def}_\Lambda\Pi(\varphi)$ which is compatible with the transitive closure of $\succ_d$, that is, $\succ_d^+ \subseteq \succ_D$. Such an ordering can always be found. For this, it was important to introduce different predicate symbols for positive and negative subformulae associated with relational subformulae.

By definition, a predicate symbol Q is *associated with a function symbol* f, written

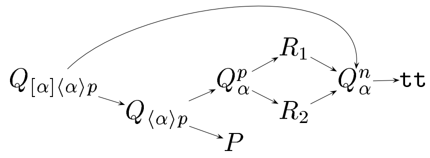

FIG. 1. Dependency among predicate symbols for (7.1)

Q_f, if there is a clause $\neg Q(x) \vee \mathcal{R}(x, f(x))$ in N. Define a measure μ as follows:

$$\mu(C) = \begin{cases} (\mathcal{P}, \mathcal{P}) & \text{if } C = \mathcal{P}(s) \\ (Q, \mathcal{R}) & \text{if } C = \mathcal{R}(s,t) \\ (\mathtt{tt}, \mathtt{tt}) & \text{if } C = \emptyset, \end{cases}$$

where Q is the predicate symbol associated with the leading function symbol of the maximal term in $\{s, t\}$. For example, the measure of a clause $R(s, f(s))$ is (Q_f, R). Complexity measures are compared by the lexicographic combination $\succ_c = (\succ_D, \succ_D)$. Now, it is routine to verify that any inference step from positive premises C_1, C_2 by resolution or factoring will produce a clause D such that $\mu(C_1) \succ_c \mu(D), \mu(C_2) \succ_c \mu(D)$. For example, for the inference step

$$\frac{Q_\psi(s)^+ \qquad \mathcal{R}(s, f(s))^+ \qquad \neg Q_\psi(x)^+ \vee \neg\mathcal{R}(x,y)^+ \vee \mathcal{P}(y)}{\mathcal{P}(f(s))}$$

$\mu(\mathcal{P}(f(s))) = (\mathcal{P}, \mathcal{P})$, $\mu(Q_\psi(s)) = (Q_\psi, Q_\psi)$, and $\mu(\mathcal{R}(s, f(s))) = (Q_f, \mathcal{R})$, where Q_f is the symbol introduced for a diamond formula ϕ, say. ϕ cannot occur at a higher modal depth than ψ, which is a box formula. Hence, it follows that $Q_f (= Q_\phi) \succ_D \mathcal{P}$. Consequently, $\mu(\mathcal{R}(s, f(s))) \succ_c \mu(\mathcal{P}(f(s)))$. Since $Q_\psi \succ_D \mathcal{P}$, by definition, we also have that $\mu(Q_\psi(s)) \succ_c \mu(\mathcal{P}(f(s)))$. For

$$\frac{Q_\psi(s)^+ \qquad \neg Q_\psi(x)^+ \vee \mathcal{R}(x, f(x))}{\mathcal{R}(s, f(s))}$$

we have that $\mu(Q_\psi(s)) = (Q_\psi, Q_\psi) \succ_c (Q_\psi, \mathcal{R}) = \mu(\mathcal{R}(s, f(s)))$, because $Q_\psi \succ_D \mathcal{R}$. For the following inference, we have that $\mu(\mathcal{R}(s, f(s))) = (Q_f, \mathcal{R}) \succ_c (Q_f, Q^n_\alpha) = \mu(Q^n_\alpha(s, f(s)))$, since α occurs negatively in φ and thus $\mathcal{R} \succ_D Q^n_\alpha$.

$$\frac{\mathcal{R}(s, f(s))^+ \qquad Q^n_\alpha(x,y) \vee \neg\mathcal{R}(x,y)^+}{Q^n_\alpha(s, f(s))}$$

It follows that any derivation terminates. ∎

THEOREM 7.3

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 $R^{\mathcal{MOD}}$ 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.

PROOF. Suppose φ is an arbitrary formula of $K_{(m)}(\cap, \cup, \smile)$ and N is the associated input set. φ has at most n subformulae, and hence, the number of clauses belonging to N is $O(n)$. Also, N contains at most n different predicate symbols (roughly one for each subformula), d different unary function symbols and one constant symbol. Recall from Lemma 7.1, split derived clauses are ground unit clauses of a certain form. As the maximal term depth bound is given by the modal depth of the input formula, there are at most $O(nd^m)$ such split clauses. It follows that the number of different literals in any derivation tree is bound by $O(nd^m)$. ∎

For logics without converse, space can be conserved by adopting the common tableaux inference strategy of considering disjunctive branches and branches associated with different $\Diamond$-subformulae in turn. In addition, the inferences with definitional clauses associated with diamond subformulae need to be postponed until no other inferences with definitional clauses associated with Boolean subformulae[3] are possible. This provides a PSPACE resolution procedure for logics in-between K and $K_{(m)}(\cap, \cup)$.

Automatically Generating Models

Any saturated clause set derivable from a given set N allows for the effective construction of a model of N. In general this model will not be finite. However, for $K_{(m)}(\cap, \cup, \smile)$ models are given by a finite set of positive ground unit clauses. The proofs of the results in this subsection are slight modifications of the corresponding results in Hustadt and Schmidt [29].

Formally, a *model* of a clause set is a set I of ground atoms. The presence of an atom A in I means A is true in I, and the absence of A means $\neg A$ is true in I. In general, a clause C is true in I iff for all ground substitutions σ there is a literal L in $C\sigma$ which is true in I. Falsehood is defined dually.

LEMMA 7.4
Let φ be a $K_{(m)}(\cap, \cup, \smile)$-formula. Let N be the clausal form of $\mathrm{Def}_\Lambda \Pi(\varphi)$, and let N_∞ denote the $R^{\mathcal{MOD}}$-saturated clause set derivable from N. Let I be the set of positive ground unit clauses in N_∞. If N_∞ does not contain the empty clause then I is a model of N_∞ and N.

Now it is an easy matter to construct a modal model $\mathcal{M} = (W, R, \iota)$ for φ from I. Essentially, the set of worlds is defined by the set of ground terms occurring in I. The interpretation of relational formulae is determined by the set of R_i literals in I. For any R_i, if $R_i(s,t)$ is in I 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 I, $s \in \iota(p_i)$ (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 7.5
For any modal formula satisfiable in $K_{(m)}(\cap, \cup, \smile)$ a finite modal model can be effectively constructed on the basis of $R^{\mathcal{MOD}}$.

[3] In Boolean subformulae the outermost connective is a Boolean connective.

COROLLARY 7.6
Let L be any logic in-between K and $K_{(m)}(\cap, \cup, \smile)$. Then, L has the finite model property.

Generalisation

Results 7.2, 7.5 and 7.6 can be generalised.

THEOREM 7.7
Let L be a logic in-between K and $K_{(m)}(\cap, \cup, \smile)$. Let Δ be a finite $R^{\mathcal{MOD}}$-saturated set of clauses consisting of two kinds of split components.

(7.3) 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).

(7.4) Clauses built from one variable, finitely many function symbols (including constants), and finitely many binary predicate symbols R_j, with the restriction that (i) the argument multisets of all non-ground literals coincide, and (ii) 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:

1. Any $R^{\mathcal{MOD}}$-derivation from $N \cup \Delta$ terminates.
2. φ is unsatisfiable in $L\Delta$ iff the $R^{\mathcal{MOD}}$-saturation of $N \cup \Delta$ contains the empty clause.

PROOF. Soundness and completeness follows by the general soundness and completeness result of ordered resolution with selection (Theorem 4.1).

For the problem of termination we first consider what kind of clauses we are dealing with. Input clauses have the form (7.2), (7.3) or (7.4). To begin with we consider the saturation of all theory clauses and the subset of clauses in N which contain only binary predicate symbols. The latter have the form:

$$(') \quad \neg Q^p_\alpha(x,y)^+ \vee \mathcal{R}(x,y)\ [\vee\ \mathcal{R}(x,y)] \quad \text{and} \quad Q^n_\alpha(x,y) \vee \neg\mathcal{R}(x,y)^+\ [\vee\ \neg\mathcal{R}(x,y)^+].$$

We will prove that (non-empty maximally split) inferred clauses include ground unit clauses (more precisely, clauses of the form $[\neg]\mathcal{R}(s,t)$, where s and t are ground terms), and clauses which are specified by (7.4), except that they may be defined over the given R_j symbols and the introduced $Q^{p/n}_\alpha$ symbols. Call such clauses $(7.4)^+$. Let $\mathcal{K}$ denote the class of clauses ($'$), (7.3), $(7.4)^+$ and ground unit clauses, defined over a finite signature. W.l.o.g. we consider only maximally split clauses.

Claim 1: Inferences with clauses satisfying (7.3) and clauses ($'$) or ground unit clauses produce ground clauses only. Assume C is a (7.3) clause. C participates in inference steps as a negative premise and the only potential partners are ground clauses. As at least one of the eligible literals in C is a guard literal the result of such an inference step is a ground resolvent.

Claim 2: Inferences with clauses satisfying (7.3) and $(7.4)^+$ produce clauses with ground or $(7.4)^+$ split components. The proof is not difficult. Observe that the conclusion of a resolution step in $R^{\mathcal{MOD}}$ is always a positive clause.

Claim 3: Inferences with $(7.4)^+$ clauses and $(')$ clauses or ground unit clauses produce either ground clauses or $(7.4)^+$ clauses. First consider any positive $(7.4)^+$ clause C. Clearly, the factor of C is again a clause satisfying $(7.4)^+$. Now consider the possibilities for resolution inferences with C.

1. Assume C is resolved with a clause of the form $\neg Q_\psi(x)^+ \vee \neg\mathcal{R}(x,y)^+ \vee \mathcal{P}(y)$. The other positive premise besides C will be a ground unit clause $Q_\psi(s)^+$. Regardless of whether C is ground or not the conclusion will be a ground clause (because the single variable that may occur in C will be instantiated with a ground term).

2. Another possibility is that C is resolved with a clause $Q^n_\alpha(x,y) \vee \neg\mathcal{R}(x,y)^+$, or a clause $Q^n_\alpha(x,y) \vee \neg\mathcal{R}(x,y)^+ \vee \neg\mathcal{R}'(x,y)^+$. In the first case the resolvent is a variation of C, namely C with the predicate symbol of the eligible literal replaced by Q^n_α and possibly the arguments exchanged. In the second case the form of the resolvent depends on the second positive premise. If the second premise is ground then the resolvent will also be ground (because if C is not ground then the single variable of C will be instantiated with a ground term). The second premise C' may be a $(7.4)^+$ clause which is not ground. In this case a resolution step is only possible if the multisets of arguments of the eligible literals of C and C' are identical. It follows that any resolvent satisfies the conditions of $(7.4)^+$. Notice that no term depth growth occurs.

3. The third possibility is that C is resolved with a clause (7.3). This possibility is covered by Claim 2.

Second, consider the case that C is a non-positive $(7.4)^+$ clause. C can only be a negative premise in a resolution inference step. The only resolution partners are ground unit clauses and positive $(7.4)^+$ clauses. In the first case the conclusion is a ground clause, and in the latter case the conclusion is again a $(7.4)^+$ clause.

Claim 4: Inferences with input clauses $(')$ and ground unit clauses produce ground clauses. The argument is similar as for Lemma 7.4.

Claims 1 to 4 prove that the class $\mathcal{K}$ is closed under inferences in $R^{\mathcal{MOD}}$. Now the saturation Δ' of Δ and the set of $(')$ clauses in N is a subset of $\mathcal{K}$. Δ' is bounded because inferred clauses contain at most two variables and there is no increase of the term depth.

We now establish that, conclusions of further inferences are ground. No inferences are possible between theory clauses satisfying condition (7.3) and clauses in N. Inferences with clauses not in Δ' are with ground clauses, and produce ground clauses. Similarly, inferences with clauses $(7.4)^+$ and clauses not in Δ' are with ground clauses, and produce ground clauses. The remaining inferences are as in Lemma 7.1. It follows that non-empty split ground conclusions have the form

$$\mathcal{P}(s), \quad (\neg)\mathcal{R}(s,f(s)), \quad (\neg)\mathcal{R}(s,s) \quad \text{or} \quad (\neg)\mathcal{R}(b,c).$$

Termination of any derivation from $N \cup \Delta$ is now shown as follows. Let $\mathtt{T}$, $\mathtt{tt}$ be new symbols which do not occur in either N or Δ. Again, we use a dependency relation $\succ_d$ on the predicate symbols. It is defined almost as in the proof of Theorem 7.2, but with subtle differences: $S_1 \succ_d S_2$, if there is a definition $\psi \to \phi$ in $\mathrm{Def}_\Lambda\Pi(\varphi)$ such that S_1 occurs in ψ and S_2 occurs in ϕ. In addition, all relational symbols R_j which do not occur in N are smaller than any unary predicate symbols. Let $\mathtt{T}$ be the largest symbol with respect to $\succ_d$, and $\mathtt{tt}$ the smallest symbol. As before, let $\succ_D$ be any ordering compatible with the transitive closure of $\succ_d$. In addition, define

$$\Gamma(T) = \{(\neg)R(s,t) \mid s,t \in T \text{ and } R \text{ is a binary predicate symbol}\}.$$

Let N_C denote the set of clauses derived prior to C, and C itself. Now, define a measure on (a subset of) clauses in a derivation by:

$$\mu(C) = \begin{cases} (\mathcal{P}, \emptyset) & \text{if } C = \mathcal{P}(s) \\ (Q, \Gamma(\{s,t\}) \setminus N_C) & \text{if } C = (\neg)\mathcal{R}(s,t) \\ (\mathtt{tt}, \emptyset) & \text{if } C = \emptyset, \end{cases}$$

where Q is the predicate symbol associated with the leading function symbol of the maximal term in $\{s,t\}$, whenever such a symbol exists, and $\mathtt{T}$ otherwise. Here, maximality is with respect to the proper subterm ordering. The ordering on the complexity measures $\succ_c$ of positive premises and conclusions is defined to be the lexicographic combination of $\succ_D$ and the proper superset relationship. This ordering is well-founded. Now we need to verify that split ground conclusions are strictly smaller than their positive premises, which is routine. Termination follows. ∎

THEOREM 7.8
Let L and Δ be as in the previous theorem. For any modal formula satisfiable in $L\Delta$ a finite modal model can be effectively constructed on the basis of $R^{\mathcal{MOD}}$.

PROOF. The construction of a modal model $\mathcal{M}$ is as above from the set of ground unit literals in the saturation of $N \cup \Delta$. It remains to consistently complete $\mathcal{M}$ in accordance with the background theory Δ. This is always possible because Δ contains no clauses requiring the creation of new worlds. For example, if $\Delta = \{R_i(x, f(x))\}$ then add $(s,s) \in R(r_i)$ for each dead end world s. Only propositional literals will be true in s. ∎

COROLLARY 7.9
Let L and Δ be as in the previous theorem. Then, $L\Delta$ has the finite model property.

Which extended modal logics satisfy the conditions of Theorem 7.7? Relational frame properties which can be described by the above clausal form 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,

$$\forall x \exists y \, \neg R(x,y), \quad \forall x \exists y \, (R(x,y) \vee R(y,x)), \quad \text{or} \quad \forall xy \, (R(x,y) \rightarrow R(x,x)).$$

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 also cover a variety of description logics, for example, $\mathcal{ALC}$ endowed with role conjunction, role disjunction and inverse roles, acyclic TBox statements, and both concept and role ABox statements.

By refining $R^{\mathcal{MOD}}$ with an ordering restriction which would prefer to resolve upon literals containing functional terms of the theory clauses in Δ we expect that the above decidability result can be improved considerably.

Finally, let us look at a sample derivation, in Figure 2, and make a few observations. R is assumed to be reflexive, for otherwise not many inference steps are possible. First, notice how the correspondence to modal subformulae is retained during inference in

1.	$Q_\varphi(a)$	
2.	$\neg Q_\varphi(x)^+ \vee Q_\Box(x)$	
3.	$\neg Q_\varphi(x)^+ \vee Q_\neg(x)$	
4.	$\neg Q_\Box(x)^+ \vee \neg R(x,y)^+ \vee Q_\vee(y)$	
5.	$\neg Q_\vee(x)^+ \vee P(x) \vee Q_\Diamond(x)$	
6.	$\neg Q_\Diamond(x)^+ \vee R(x, f(x))$	
7.	$\neg Q_\Diamond(x)^+ \vee P(f(x))$	
8.	$\neg Q_\neg(x)^+ \vee \neg P(x)^+$	
9.	$R(x,x)$	
10.	$Q_\Box(a)$	[1, 2]
11.	$Q_\neg(a)$	[1, 3]
12.	$Q_\vee(a)$	[4, 9, 10]
13.	$P(a) \vee Q_\Diamond(a)$	[5, 12]

14.1.1.	$P(a)$	[13, split]	14.2.1.	$Q_\Diamond(a)$	[13, split]
14.1.2.	$\emptyset$	[8, 11, 14.1.1]	14.2.2.	$R(a, f(a))$	[6, 14.2.1]
			14.2.3.	$P(f(a))$	[7, 14.2.1]
			14.2.4.	$Q_\vee(f(a))$	[4, 10, 14.2.2]
			14.2.5.	$P(f(a)) \vee Q_\Diamond(f(a))$	[5, 14.2.4]

14.2.5.1.1.	$P(f(a))$	[14.2.5, split]	14.2.5.1.2.	$Q_\Diamond(f(a))$	[14.2.5, split]

FIG. 2. Derivation tree for testing the satisfiability of $\varphi = \Box(p \vee \Diamond p) \wedge \neg p$ in KT.

$R^{\mathcal{MOD}}$. For example, 14.1.1 and 14.2.1 say that p and $\Diamond p$ are true in the initial world a, 14.2.2, 14.2.3 and 14.2.4 say that p and $p \vee \Diamond p$ are true in a successor world of a. Second, notice the similarity of this derivation to the derivation of a classical tableaux procedure. This connection will be formally discussed in the next section.

8 Tableaux Calculi

Selection refinements of resolution (and hyperresolution) are closely related to standard modal tableaux calculi and description logic systems [13, 27, 28, 29]. In this section, we exploit this connection and present tableaux calculi for the modal logic $K_{(m)}(\cap, \cup, \smile)$, and logics below it. These calculi resemble and enhance those commonly used in description logic systems [22, 21]. We also investigate the relationship between our selection-based resolution procedure and single-step prefixed tableaux calculi.

Tableaux Calculi for Subsystems of $K_{(m)}(\cap, \cup, \smile)$

A *tableaux* 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

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

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

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

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

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.

FIG. 3. Tableaux expansion rules for $K_{(m)}(\cap,\cup,\smile)$.

For $K_{(m)}$:	$(\bot),(\wedge),(\vee),(\Diamond),(\Box)$
For $K_{(m)}(\smile)$:	$(\bot),(\wedge),(\vee),(\Diamond),(\Box),(\smile),(\smile_I)$
For $K_{(m)}(\cap)$:	$(\bot),(\wedge),(\vee),(\Diamond),(\Box),(\wedge^r),(\wedge^r_I)$
For $K_{(m)}(\cup)$:	$(\bot),(\wedge),(\vee),(\Diamond),(\Box),(\vee^r),(\vee^r_I)$
For $K_{(m)}(\cap,\cup)$:	$(\bot),(\wedge),(\vee),(\Diamond),(\Box),(\wedge^r),(\wedge^r_I),(\vee^r),(\vee^r_I)$
$\vdots$	$\vdots$

FIG. 4. Tableaux calculi for logics in-between $K_{(m)}$ and $K_{(m)}(\cap,\cup,\smile)$.

$\frac{X}{X_1\mid\ldots\mid X_n}$ 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. In the following we assume φ is a formula in negation normal form.

Figure 3 lists the expansion rules for the logic $K_{(m)}(\cap,\cup,\smile)$, while for any logic L in-between K and $K_{(m)}(\cap,\cup,\smile)$ the expansion rules are given by appropriate subsets, see Figure 4. 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. Note that only the disjunction rules are "don't know" nondeterministic and require the use of backtracking.

To avoid unnecessary duplication and superfluous inferences we define a notion of redundancy which is in the spirit of Bachmair and Ganzinger [3]. A labelled formula F is redundant in a node if the node contains labelled formulae $F_1,\ldots,F_n$ (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 [3, 4] may be chosen. The application of a rule is redundant if 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 8.1
A formula φ is satisfiable in $K_{(m)}(\cap, \cup, \smile)$ iff a tableaux containing a branch $\mathcal{B}$ can be constructed with the rules of Figure 3 such that $\mathcal{B}$ does not contain the falsum ($s : \perp$ for some s) and each rule application is redundant.

PROOF. By soundness, completeness and termination of the selection refinement $R^{\mathcal{MOD}}$ (Theorem 7.2), and the observation that the tableaux rules are macro inference steps of $R^{\mathcal{MOD}}$ on the set

$$N = N' \cup \{\neg Q^p_\alpha(x, y)^+ \vee Q^n_\alpha(x, y) \mid \alpha \text{ is a non-atomic relational formula in } \varphi\},$$

where N' is the clausal form of $\mathrm{Def}_\Lambda\Pi(\varphi)$, and Λ is as defined at the beginning of the previous section. For this extended N the termination argument is the same as in Theorem 7.2.

Define a mapping h' from labelled formulae to ground unit clauses by (h' is in fact a bijection)

$$\begin{aligned} h'(s : \psi) &= h(\psi)(h(s)) \\ h'((s, t) : \alpha) &= h(\alpha)(h(s), h(t)), \end{aligned}$$

where ψ denotes a modal formula, α a relational formula. h is defined by: $h(p_i) = P_i$, $h(r_j) = R_j$, $h(\psi) = Q_\psi$, $h(\alpha) = Q^p_\alpha$, $h(a) = a$, and $h(t) = f_{\langle\alpha\rangle\psi}(h(s))$ where $s : \langle\alpha\rangle\psi$ is the formula for which t was introduced and $f_{\langle\alpha\rangle\psi}$ is the Skolem function associated with $\langle\alpha\rangle\psi$.

The $R^{\mathcal{MOD}}$-derivation corresponding to an application of the $(\Diamond)$-rule is: from $Q_{\langle\alpha\rangle\psi}(h(s))$, $\neg Q_{\langle\alpha\rangle\psi}(x)^+ \vee Q^p_\alpha(x, f(x))$ and $\neg Q_{\langle\alpha\rangle\psi}(x)^+ \vee Q_\psi(f(x))$, derive the units $Q^p_\alpha(h(s), f(h(s)))$ and $Q_\psi(f(h(s)))$ in two resolution steps. For $(\smile_I)$ the resolvent of $Q^n_\alpha(h(s), h(t))$ (or $Q^p_\alpha(h(s), h(t))$ and $\neg Q^p_\alpha(x, y)^+ \vee Q^n_\alpha(x, y)$) and $Q^n_{\alpha^\smile}(x, y) \vee \neg Q^n_\alpha(y, x)^+$, is $Q^n_{\alpha^\smile}(h(t), h(s))$. Similarly, for the other rules.

Apart from factoring there are no inference steps in $R^{\mathcal{MOD}}$ which are not involved in some macro inference step. Due to the fact that all positive premises are ground and thus subject to the application of splitting, factoring is not needed for completeness, and is thus optional. ∎

COROLLARY 8.2
The appropriate subsets (see Figure 4) of the rules from Figure 3 provide sound, complete and terminating tableaux calculi for logics in-between K and $K_{(m)}(\cap, \cup, \smile)$.

An immediate consequence of Theorem 7.5 is:

COROLLARY 8.3
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 appropriate tableaux calculus for L.

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

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

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

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

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

FIG. 5. Single step prefixed tableaux expansion rules for subsystems of *S5*.

Simulation of Single-Step Prefixed Tableaux

We distinguish between two notions of polynomial simulation (or p-simulation). By definition, a proof system $\mathcal{A}$ *p-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. We also say 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 [6], 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_2 of φ in $\mathcal{B}$, there exists a derivation D_1 of φ in $\mathcal{A}$ such that the number of applications of inference rules in D_1 is polynomially bounded by the number of applications of inference rules in D_2. This can be achieved by showing that there exists a number n such that each application of an inference rule in D_1 corresponds to at most n applications of inference rules in D_2. It follows that the length of D_2 is polynomially bounded by the length of D_1. We call this a *step-wise simulation* of $\mathcal{B}$ by $\mathcal{A}$. Note that a step-wise simulation is independent of whether the considered derivations are proofs or not.

The single-step prefixed tableaux calculi of Massacci [31, 33] for subsystems of *S5* are defined by Figures 5 and 6. (Remember *KT* = *KDT*, *S4* = *KT4*, *KB4* = *KB5*, *S5* = *KTB4* = *KDB4* = *KT5*.) 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. Tableaux 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 expansion rules. The prefixes in the expansion rules, except for $\sigma.n$ of the $(\Diamond)$-rule, are assumed to be present on the current branch.

THEOREM 8.4 (Massacci [31, 33], Goré [19])
Let $\Sigma \subseteq \{D, T, B, 4, 5\}$. A formula φ is satisfiable in a logic $K\Sigma$ iff a tableaux

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

FIG. 6. Tableaux calculi for subsystems of *S5*.

containing a branch $\mathcal{B}$ can be constructed by the tableaux calculus for $K\Sigma$ such that $\mathcal{B}$ does not contain the falsum and further rule applications are redundant.

The first-order background theories for the different axiom schemas are determined by the following.

$$\begin{array}{ll} T_K = \emptyset \quad T_{KD} = \{R(x, f(x))^+\} & T_{KT} = \{R(x,x)^+\} \\ T_{KB} = \{\neg R(x,y)^+ \vee R(y,x)\} & T_{K4} = \{\neg R(x,y)^+ \vee \neg R(y,z) \vee R(x,z)\} \\ T_{K5} = \{\neg R(x,y) \vee \neg R(x,z) \vee R(y,z), \neg R(x,y) \vee R(y,y)\} & \end{array}$$

For modal logics closed under more than one additional axiom schema the background theories are defined by the union of the corresponding clause sets, for example, $T_{KD4} = T_{KD} \cup T_{K4}$.

Observe that for 4 and 5 only certain negative literals will be selected in the theory clauses. In the case of 5 we do not select any literal.

THEOREM 8.5
Let $\Sigma \subseteq \{D, T, B, 4, 5\}$. Resolution p-simulates derivations of single step prefix tableaux for $K\Sigma$.

PROOF. Suppose we are interested in the satisfiability of the modal formula φ. We will show that $R^{\mathcal{MOD}}$ p-simulates single step prefix tableaux step-wise.

Similar as in the proof of Theorem 8.1 define a mapping (bijection) h' from prefixed formulae to ground unit clauses by $h'(\sigma : \psi) = h(\psi)(h(\sigma))$, where h is defined by: $h(p_i) = P_i$, $h(r_j) = R_j$, $h(\psi) = Q_\psi$ for ψ a modal subformula of φ, $h(1) = a$, and $h(\sigma.n) = f_{\Diamond\psi}(h(\sigma))$ where $\Diamond\psi$ is the formula for which n was introduced and $f_{\Diamond\psi}$ is the Skolem function associated with $\Diamond\psi$. For example, the unit clause associated (by h') with the formula $1 : \varphi$ contained in the root node is $Q_\varphi(a)$.

Now show that each tableaux inference step can be simulated by a constant number of $R^{\mathcal{MOD}}$-inference steps. For instance, the derivation of $\perp$ by the clash rule corresponds to one resolution inference step applied to $Q_\psi(h(\sigma))$, $Q_{\neg\psi}(h(\sigma))$ and $\neg Q_{\neg\psi}(x)^+ \vee \neg Q_\psi(x)^+$, which generates the empty clause. For the simulation of the application of the $(\Diamond)$ rule to $\sigma : \Diamond\psi$ we may assume that $Q_{\Diamond\psi}(h(\sigma))$ is present in the clauses set. Also present are the definitional clauses $\neg Q_{\Diamond\psi}(x)^+ \vee R(x, f(x))$, and $\neg Q_{\Diamond\psi}(x)^+ \vee Q_\psi(f(x))$. Then an application of the $(\Diamond)$ rule corresponds to performing two resolution inference steps producing $R(h(\sigma), f(h(\sigma)))$ and $Q_\psi(f(h(\sigma)))$. The term $f(h(\sigma))$ corresponds to the new prefix $\sigma.n$. The interested reader may fill in the details for the other rules, see also [29]. ∎

For the modal logics $K\Sigma$ with $\Sigma \subseteq \{D, T, B\}$ there is a near bisimulation between the tableaux calculi and $R^{\mathcal{MOD}}$. If factoring rules are added to the tableaux calculi then tableaux p-simulates also derivations of the selection-based resolution refinement. It follows that:

THEOREM 8.6
$R^{\mathcal{MOD}}$ p-simulates search in single step prefix tableaux for $K\Sigma$ with $\Sigma \subseteq \{D, T, B\}$.

This is not true for logics in which 4 and 5 are theorems. For 4 and 5 termination in single step prefixed tableaux is ensured by a loop checking mechanism [31, 33]. Once a loop is detected in a branch no further rules are applied. In $R^{\mathcal{MOD}}$ further inference steps will be performed. To prevent this we have to provide the means by which the resolution procedure can recognise the redundancy of further inference steps. This may possibly be realised by soft typing [16] or some form of blocking which is analogous to loop checking [27].

In this section we have focussed on single-step prefixed tableaux calculi, but this choice is arbitrary. Our technique can also be applied for obtaining simulation results of modal tableaux calculi with implicit or explicit accessibility relation and analytic modal KE tableaux [25, 32], or even sequent proof systems. Simulation results of tableaux calculi for description logics by resolution can be found in Hustadt and Schmidt [27, 28].

9 Concluding Remarks

The approach purported in this overview paper is that modal logics can be seen to be fragments of first-order logic and inference systems for modal logics can be developed and studied within the framework of first-order resolution. Several issues were considered. In particular, we have focussed on the decision problem for a range of expressive extended modal logics and have described resolution procedures of varying nature. We have looked at using resolution methods for automatically generating models. Exploiting the link between selection-based resolution and tableaux methods, we have proposed a new tableaux calculus for multi-modal logics defined over relations closed under union, intersection and converse. And, we have presented simulation results which give us an understanding of modal tableaux methods in the wider context of first-order logic and resolution.

Some important modal logics for which we have not presented a decision procedure are 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 [15].

Another possibility is to use the resolution procedures described in this paper 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.

Acknowledgements

We thank the referees for valuable and detailed comments.

References

[1] H. Andréka, I. Németi, and J. van Benthem. Modal languages and bounded fragments of predicate logic. *J. of Philosophical Logic*, 27(3):217–274, 1998.

[2] H. Andréka, J. van Benthem, and I. Németi. Back and forth between modal logic and classical logic. *Bull. IGPL*, 3(5):685–720, 1995.

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

[4] L. Bachmair and H. Ganzinger. Resolution theorem proving. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier, 2000. To appear.

[5] C.-L. Chang and R. C.-T. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Computer Science Classics Series. Academic Press, New York, 1973.

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

[7] H. de Nivelle. *Ordering Refinements of Resolution*. PhD thesis, Delft University of Technology, 1995.

[8] H. de Nivelle. An overview of resolution decision procedures. In M. Faller and S. Kaufmann, editors, *Formalizing the Dynamics of Information*. 1998.

[9] H. de Nivelle. A resolution decision procedure for the guarded fragment. In C. Kirchner and H. Kirchner, editors, *Automated Deduction—CADE-15*, volume 1421 of *Lecture Notes in Artificial Intelligence*, pages 191–204. Springer, 1998.

[10] H. de Nivelle. Bliksem, 1999. `http://www.mpi-sb.mpg.de/~bliksem`.

[11] M. de Rijke. A system of dynamic modal logic. *J. of Philosophical Logic*, pages 109–142, 1998.

[12] 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.

[13] C. G. Fermüller, A. Leitsch, U. Hustadt, and T. Tammet. Resolution theorem proving. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier, 2000. To appear.

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

[15] H. Ganzinger, U. Hustadt, C. Meyer, and R. A. Schmidt. A resolution-based decision procedure for extensions of K4. In K. Segerberg, M. Zakharyaschev, M. de Rijke, and H. Wansing, editors, *Advances in Modal Logic, Volume 2*, Lecture Notes. CSLI Publications, Stanford, 1999. To appear.

[16] 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. Springer, 1997.

[17] 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. Plenum Press, 1990.

[18] 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.

[19] R. Goré. Tableau methods for modal and temporal logics. In M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, editors, *Handbook of Tableau Methods*. Kluwer, 1999.

[20] E. Grädel. On the restraining power of guards. To appear in the *J. of Symbolic Logic*, 1998.

[21] B. Hollunder and W. Nutt. Subsumption algorithms for concept languages. Research Report RR-90-04, Deutsches Forschungszentrum für Künstliche Intelligenz, Saarbrücken, Germany, 1990.

[22] I. Horrocks. Using an expressive description logic: FaCT or fiction? In A. Cohn, L. Schubert, and S. Shapiro, editors, *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR-98)*, pages 636–647. Morgan Kaufmann, 1998.

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

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

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

[26] U. Hustadt and R. A. Schmidt. Maslov's class K revisited. In H. Ganzinger, editor, *Automated Deduction—CADE-16*, volume 1632 of *Lecture Notes in Artificial Intelligence*, pages 172–186. Springer, 1999.

[27] 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 Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 110–115. Morgan Kaufmann, 1999.

[28] 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 192–206. Springer, 2000.

[29] U. Hustadt and R. A. Schmidt. Using resolution for testing modal satisfiability and building models. To appear in the *SAT 2000* Special Issue of *J. of Automated Reasoning*, 2000.

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

[31] F. Massacci. Strongly analytic tableaux for normal modal logics. In A. Bundy, editor, *Automated Deduction—CADE-12*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 723–737. Springer, 1994.

[32] F. Massacci. Simplification: A general constraint propagation technique for propositional and modal tableaux. In H. de Swart, editor, *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX'98, Oisterwijk, The Netherlands, Proceedings*, volume 1397 of *Lecture Notes in Artificial Intelligence*, pages 217–231. Springer, 1998.

[33] F. Massacci. Single step tableaux for modal logics: Computational properties, complexity and methodology. *J. of Automated Reasoning*, 2000. To appear.

[34] W. McCune. *Otter*, 1995. `http://www.mcs.anl.gov/AR/otter/`.

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

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

[37] H. J. Ohlbach, A. Nonnengart, and D. Gabbay. Encoding two-valued non-classical logics in classical logic. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier, 2000. To appear.

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

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

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

Received February 14, 2000