A Resolution Decision Procedure for the Guarded Fragment

Hans de Nivelle

Centrum voor Wiskunde en Informatica,
PO BOX 94079, 1090 GB Amsterdam,
the Netherlands,
email: nivelle@cwi.nl

Abstract. We give a resolution based decision procedure for the guarded fragment of [ANB96]. The relevance of the guarded fragment lies in the fact that many modal logics can be translated into it. In this way the guarded fragment acts as a framework explaining the nice properties of these modal logics. By constructing an effective decision procedure for the guarded fragment we define an effective procedure for deciding these modal logics.

1 Introduction

The guarded fragment was inspired in [ANB96], (see also [Benthem96]) by the following observations: **(1)** Many propositional modal logics have very good properties, they are decidable, have the finite model property, and interpolation. **(2)** These modal logics can be translated into first order logic, using a standard (relational) translation based on the Kripke frames:

$$\Box A \Rightarrow \forall s' R(s,s') \rightarrow \cdots \qquad \Diamond A \Rightarrow \exists s' R(s,s') \wedge \cdots$$

The fragment of first order formulae that can be a translation of a modal formula must also have these properties. This leads to the following question: What makes this set of translations of modal formulae so nice? One explanation could be the fact that modal logic translates into the 2-variable fragment, which is decidable. This is not sufficient. The logic K can be translated into the 2-variable fragment, but most other modal logics cannot. Also the 2-variable fragment lacks interpolation, although it is decidable.
The guarded fragment is based on the observation that in the translations all quantifiers are *conditional* in an accessibility condition, i.e. they all have the form: for all worlds s' for which $R(s,s')$, something holds in s'. This leads to a definition of the *guarded fragment* in which all universal quantifiers occur as $\forall \overline{x}(G(\overline{x},\overline{y}) \rightarrow \Phi(\overline{x},\overline{y}))$, where G is an atom. It turns out that this fragment, although it still cannot explain all modal logics, has good model theoretic properties. There are

C. Kirchner and H. Kirchner (Eds.): Automated Deduction, CADE-15
LNAI 1421, pp. 191–204, 1998.

more perspectives for generalization (see [Benthem97]), than from the 2-variable fragment, as the 3-variable fragment is already undecidable.
Among the logics that can be translated into the guarded fragment are the modal logics $K, D, T, S5$, many arrow logics, and weak predicate logics, (see [Benthem96]). Logic $S4$ does not fit, because of the transitivity axioms.
In this paper we develop a resolution decision procedure for the guarded fragment. We define guarded clauses, and show that first order guarded formulae can be translated into sets of guarded clauses. After that we show that sets of guarded clause sets are decidable using techniques that are standard in the field of resolution decision procedures. The restriction of resolution that has to be used is based on an *ordering refinement*. All of the major theorem provers (SPASS, [Wbach96], OTTER [McCune95], and Gandalf, [ATP97]) support orderings, although not exactly the one that we need. We have implemented our strategy as an option in a general purpose, resolution theorem prover.
In [Ohlbach88a] and [Ohlbach88b], the *functional* translation of modal logics is introduced, as opposed to the *relational* translation which is the one that we use. In the functional translation, the accessibility relation is translated into many function symbols, instead of one relation symbol. It is argued there that this has the advantage of resulting in a decision procedure, and that relational translation does not result in a decision procedure. We show that it is possible to obtain a decision procedure, using the relational translation.
Another approach to theorem proving in modal logics can be found in [FarHerz88] and [EnjFar89]. Instead of translating the modal formula, resolution rules are defined, that work directly in the modal logic. The rules are complicated and for each modal logic, a new calculus has to be designed. In [Nivelle93], [Nivelle92], these problems were partially overcome. A generic approach to resolution in propositional modal systems was defined there, but the rules are still complicated, and computationally costly. A resolution based decision procedure based on the guarded fragment is generic, and the effort of implementation is low.

2 The Guarded Fragment

In this section we briefly introduce the guarded fragment:

Definition 1. *A term is* functional *if it is not a constant, nor a variable. The* guarded fragment *of first order logic ($\mathcal{GF}$) is recursively built up as follows:*

1. $\top$ *and* $\bot$ *are in* $\mathcal{GF}$.
2. *If A is an atom, such that none of its arguments is is functional, then* $A \in \mathcal{GF}$.
3. *If* $A \in \mathcal{GF}$, *then* $\neg A \in \mathcal{GF}$.
4. *If* $A, B \in \mathcal{GF}$, *then* $A \vee B$, $A \wedge B$, $A \rightarrow B$, $A \leftrightarrow B \in \mathcal{GF}$.
5. *If* $A \in \mathcal{GF}$ *and a is an atom, for which* **(a)** *all arguments of a are non-functional,* **(b)** *every free variable of A is among the arguments of a, then* $\forall \overline{x}(a \rightarrow A) \in \mathcal{GF}$, *for every sequence of variables* $\overline{x}$.

6. *If $A \in \mathcal{GF}$ and a is an atom, for which* (**a**) *every argument of a is a non-functional,* (**b**) *every free variable of A is among the arguments of a, then $\exists \overline{x}(a \wedge A) \in \mathcal{GF}$.*

The atoms a are called the guards. *The guards may have repeated arguments, and they do not need to occur in some fixed order. So $a(y, x, x, y)$ is allowed as guard. Each occurrence of a quantifier can have a different guard.*

Example 1. The formulae $\forall x(a(x,y) \rightarrow b(x,y))$ and $\exists x(a(x,y) \wedge b(x,y))$ are in $\mathcal{GF}$. Also the formulae $\forall xy(a(x,y) \rightarrow b(x,y))$ and $\exists xy(a(x,y) \wedge b(x,y))$. The formulae $\forall x(a(x) \rightarrow b(x,y))$ and $\exists x(a(x) \wedge b(x,y))$ are not. The formula

$$\forall x(a(x,y) \rightarrow \forall z(b(y,z) \rightarrow c(y,z) \wedge d(y,z)))$$

is guarded. The formula

$$\exists x(a(x,y) \wedge \forall z(b(y,z) \rightarrow c(y,z) \vee d(x,z)))$$

is not guarded. The formula

$$\forall x_1 x_2 x_3 [R(x_1,x_2) \rightarrow R(x_2,x_3) \rightarrow R(x_1,x_3)],$$

which expresses transitivity, is not guarded. The modal formula $\Diamond(\Box a \wedge b)$ translates into $\exists x[R(c,x) \wedge (\ \forall y R(x,y) \rightarrow a(y)\ ) \wedge b(y)]$, which is guarded, as we promised in the introduction. (c is the present world).

3 Resolution

We briefly review some notions:

Definition 2. *We assume a fixed, infinite set of function/constant symbols F, a fixed, infinite set of predicate/propositional symbols P, and a fixed, infinite set of variables V. The set of* terms *is recursively defined as follows:* (**1**) *A variable is a term.* (**2**) *If $t_1, \ldots, t_n$, with $n \geq 0$, are terms, and $f \in F$, then $f(t_1, \ldots, t_n)$ is a term. If $t_1, \ldots, t_n$, with $n \geq 0$, are terms, and $p \in P$, then $p(t_1, \ldots, t_n)$ is an* atom. *A* literal *is an atom A, or its negation $\neg A$. Atoms of the form A are called* positive. *Atoms of the form $\neg A$ are called* negative. *A* clause *is a finite set of literals. A term that contains no variables is called* ground. *A term of the form c is called* constant. *A term of the form $f(t_1, \ldots, t_n)$, with $n > 0$, is called* functional.

Definition 3. *We define some complexity measures for atoms/clauses/literals: Let A be an atom/term. The* depth *of A is recursively defined as follows:* (**1**) *If A is a variable, then* $\mathrm{Depth}(A) = 1$. (**2**) $\mathrm{Depth}(f(t_1, \ldots, t_n))$ *equals the maximum of* $\{1, 1 + \mathrm{Depth}(t_1), \ldots, 1 + \mathrm{Depth}(t_n)\}$. *The depth of a literal equals the depth of its atom. The depth of a clause c equals the maximal depth of a literal in c, or -1 for the empty clause. The depth of a set of clauses equals the depth of the deepest clause.*

Let A be an atom/term. The vardepth *of A is recursively defined as follows:* **(1)** *If A is ground, then* $\mathrm{Vardepth}(A) = -1$, **(2)** *If A is a variable, then* $\mathrm{Vardepth}(A) = 0$, **(3)** *Otherwise* $\mathrm{Vardepth}(f(t_1, \ldots, t_n)$ *equals the maximum of* $\{1 + \mathrm{Vardepth}(t_1), \ldots, 1 + \mathrm{Vardepth}(t_n)\}$. *The vardepth of a literal equals the vardepth of its atom. The vardepth of a clause c equals the maximal depth of a literal in c. The vardepth of the empty clause is defined as -1. The vardepth of a set of clauses C is defined as the maximal vardepth of a clause in C.*
Let A be an atom/literal/clause. $\mathrm{Var}(a)$ *is defined as the set of variables that occur in A.*
Let A be an atom/literal/clause. $\mathrm{Varnr}(A)$ *equals the size of* $\mathrm{Var}(A)$. *If C is a set of clauses, then* $\mathrm{Varnr}(C)$ *equals maximal number of variables that occur in a clause of C.*
Let A be a literal. The complexity *of A, written as $\#A$ equals the total number of function/constant/variable occurrences in it.*

So $\mathrm{Depth}(p(X)) = 2$, and $\mathrm{Vardepth}(p(X)) = 1$. This is because the Vardepth is defined by the depth at which variable X occurs, where the Depth is defined by the depth that X creates. For a literal A holds that $\mathrm{Vardepth}(A) = -1$ implies that A is ground. $\mathrm{Vardepth}(A) = 0$ is not possible. $\mathrm{Vardepth}(A) = 1$ means that A is non-ground, but has no non-ground, functional arguments. $\mathrm{Vardepth}(A) > 1$ means that A is non-ground, and has non-ground, functional arguments.

Definition 4. *A* substitution *is a finite set of variable assignments of the form $\{V_1 := t_1, \ldots, V_n := t_n\}$, such that $V_i \neq t_i$, and $V_i = V_j \Rightarrow t_i = t_j$. The first conditions ensures non-redundancy, the second condition ensures consistency. We write $A\Theta$ for the effect of Θ on term A.*
If Θ_1 and Θ_2 are substitutions, then the composition *of Θ_1 and Θ_2 is defined as the substitution $\{v := v\Theta_1\Theta_2 \mid v \neq v\Theta_1\Theta_2\}$. We write $\Theta_1 \cdot \Theta_2$ for the composition of Θ_1 and Θ_2.*
For two literals A and B a unifier is a substitution Θ, such that $A\Theta = B\Theta$. A most general unifier *Θ is a substitution such that $A\Theta = B\Theta$, and $\forall\Theta'\ A\Theta' = B\Theta' \Rightarrow \exists\Sigma\ \Theta' = \Theta \cdot \Sigma$.*

The notion of mgu was introduced by J. A. Robinson in [Robinson65].

Definition 5. *We define the ordered resolution rule, and factorization rule: Let $\sqsubset$ be an order on literals. Let $\{A_1\} \cup R_1$ and $\{\neg\ A_2\} \cup R_2$ be two clauses, s.t.* **(1)** *$\{A_1\} \cup R_1$ and $\{\neg\ A_2\} \cup R_2$ have no variables in common,* **(2)** *for no $A \in R_1$, it is the case that $A_1 \sqsubset A$,* **(3)** *for no $A \in R_2$, it is the case that $A_2 \sqsubset A$, and* **(4)** *A_1 and A_2 have mgu Θ. Then the clause $R_1\Theta \cup R_2\Theta$ is called a* resolvent.
Let $\{A_1, A_2\} \cup R$ be a clause, such that A_1 and A_2 have an mgu Θ. The clause $\{A_1\Theta\} \cup R\Theta$ is called a factor *of $\{A_1, A_2\} \cup R$.*

It is also possible to restrict factorization by the ordering, but we prefer not to do that, since we are not certain that this improves efficiency.

4 Covering Literals

Most resolution decision procedures rely on the notion of (weakly) covering literals. The guarded fragment is no exception.

Definition 6. *A literal A is* covering *if every functional subterm t of A contains all variables of A. A literal A is* weakly covering *if every functional, non-ground subterm t of A contains all variables of A.*

Covering and weakly covering literals are typically the result of skolemization, when the prefix ends in an existential quantifier. If an atom $a(\overline{x}, y)$ in the scope of $\forall\overline{x}\exists y$ is skolemized the result equals $a(\overline{x}, f(\overline{x}))$, which is covering. If $a(\overline{x}, y)$ contains ground terms, then the result is weakly covering.
The main property of (weakly) covering literals is that they do not grow (too much) when they are unified. Theorem 1 states that when two weakly covering literals are unified, the maximal depth of a variable does not grow. Theorem 2 states that there are no new ground terms in the result, unless the result is completely ground.

Theorem 1. *Let A and B be weakly covering literals that have an mgu Θ. Let $C = A\Theta = B\Theta$. Then:* **(1)** *C is weakly covering,* **(2)** $\mathrm{Vardepth}(C) \leq \mathrm{Vardepth}(A)$, *or* $\mathrm{Vardepth}(C) \leq \mathrm{Vardepth}(B)$, *and* **(3)** $\mathrm{Varnr}(C) \leq \mathrm{Varnr}(A)$ *or* $\mathrm{Varnr}(C) \leq \mathrm{Varnr}(B)$.

For a proof, see [FLTZ93] or [Nivelle98].

Theorem 2. *Let $C = A\Theta = B\Theta$ be the most general unifier of two weakly covering literals. If C is not ground by itself, then every ground term of C occurs either in A or in B.*

For a proof see [Nivelle98] or [FLTZ93].
This shows that literals resolved upon do not grow, but it is also necessary to show that side literals can be bounded by the literals resolved upon. First we show that the side literals will be weakly covering. After that we show that they are not too deep:

Theorem 3. *Let A and B be literals and let Θ be a substitution such that* **(1)** $\mathrm{Var}(A) \subseteq \mathrm{Var}(B)$, **(2)** *$A$ is weakly covering,* **(3)** *B is weakly covering,* **(4)** *$B\Theta$ is weakly covering. Then $A\Theta$ is weakly covering.*

See [FLTZ93], or [Nivelle98] for a proof.

Lemma 1. *If* **(1)** $\mathrm{Var}(A) \subseteq \mathrm{Var}(B)$, **(2)** *$A$ is weakly covering,* **(3)** *B is weakly covering,* **(4)** $\mathrm{Vardepth}(A) \leq \mathrm{Vardepth}(B)$, *then* $\mathrm{Vardepth}(A\Theta) \leq \mathrm{Vardepth}(B\Theta)$.

5 Transformation to Clausal Normal Form

Since clauses are a restricted subset of first order logic, we need a transformation of first order logic to clausal normal form. The standard clause transformations do not work, since they would not lead to a decision procedure. We first define the notion of 'guarded' for clause sets, after that we show that a first order formula in $\mathcal{GF}$ can be effectively translated into a guarded clause set.

Definition 7. *A clause set C is called* guarded *if its clauses are guarded. A clause c is called guarded if it satisfies the following conditions:*

1. *The literals $A \in c$ are weakly covering.*
2. *If c is not ground, then there is a literal $A \in c$ with* $\mathrm{Vardepth}(A) = 1$, *such that* $\mathrm{Var}(A) = \mathrm{Var}(c)$, *and A is negative. (*$\mathrm{Vardepth}(A) = 1$ *means that all arguments of A are a constant or a variable)*
3. *If* $\mathrm{Vardepth}(A) > 1$, *then* $\mathrm{Var}(A) = \mathrm{Var}(c)$. *(*$\mathrm{Vardepth}(A) > 1$ *iff A has a non-ground argument that is functional)*

The negative literal of Condition 2 is the guard.

As a consequence every ground clause is guarded. We give a few examples:

Example 2. Clause $\{p(0, s(0)), q(s(s0))\}$ is guarded because it is ground. The clause $\{\neg\ p(X), \neg\ q(X,Y), r(f(X,Y))\}$ is guarded by $\neg\ q(X,Y)$. The clause $\{\neg\ p(X), \neg\ q(Y), r(f(X,Y))\}$ is not guarded. Adding the literal $\neg\ a(X,Y,X,X,Y)$ would make the clause guarded.

The first steps of the translation are completely standard. We define the translation operators for sets of formulae, rather than formulae. This makes it possible that an operators splits a formula into more than one formula.

Definition 8. *Let $C = \{F_1, \ldots, F_n\}$ be a set of formulae. We define* $\mathrm{Na}(C)$ *as the result of replacing $A \leftrightarrow B$ by $(\neg\ A \vee B) \wedge (\neg\ B \vee A)$, and replacing $A \rightarrow B$ by $\neg\ A \vee B$ in all the F_i.*
The negation normal form *of $C = \{F_1, \ldots, F_n\}$ is obtained by moving negations inward as far as possible, by deleting double negations, and by deleting $\top$ and $\bot$ as much as possible. We write* $\mathrm{NNF}(C)$ *for the negation normal form of C.*

The advantage of the negation normal form is that it makes the polarities of the subfomulae explicit.
In order to proceed, we need a variation of the structural transformation. Structural transformations replace certain subformulae by fresh names, together with a definition of the name. Structural translations are studied in [BFL94]. They are called structural there, because more is preserved of the structure of the formula than when the formula is factored into clausal normal form. Our structural transformation is different from the one there, as we only replace universally quantified subformulae, and in a specialized manner:

Definition 9. *Let $C = \{F_1, \ldots, F_n\}$ be a set of guarded formulae in negation normal form.* $\mathrm{Struct}_{\mathcal{GF}}(F)$ *is obtained by making the following replacements in the F_i as long as possible: As long as there is an F_i which can be written as $F_i[\forall \overline{x}(\neg\, a \vee A)]$, where F_i is not empty, (i.e. the quantifier is inside some context), let $\overline{y}$ be the free variables of a, that are not among the $\overline{x}$. So $\overline{y}$ contains exactly the free variables of $\forall \overline{x}(\neg\, a \vee A)$.*
Let α be a fresh predicate name that does not occur in an F_i. Then add

$$\forall \overline{x}\overline{y}(\neg\, a \vee \neg\, \alpha(\overline{y}) \vee A)$$

to C. (Thus increasing n by 1) Replace $F_i[\forall \overline{x}(\neg\, a \vee A)]$ by $F_i[\alpha(\overline{y})]$.

The next step is Skolemization. Skolemization is the replacement of existential quantifiers by fresh function symbols in the preceding universal quantifiers.

Definition 10. *Let $C = \{F_1, \ldots, F_n\}$ be a set of formulae in NNF. The* Skolemization *is obtained as follows:*
As long as one of the F_i contains an existential quantifier, do the following: Write $F_i = F_i[\exists y A]$, where $\exists y A$ is not in the scope of another existential quantifier. Let $x_1, \ldots, x_n$ be the universally quantified variables, in the scope of which A occurs. Replace $F_i[\exists y A]$ by $F_i[A[y := f(x_1, \ldots, x_n)]\,]$.

There exist more sophisticated ways for Skolemization leading to smaller, or more general Skolem terms ([OWbach95]), but we cannot use them here. The reason for this is that optimized Skolem translations try to remove irrelevant variables from the Skolem terms $f(v_1, \ldots, v_n)$. This would destroy Condition 3 of Definition 7.

Definition 11. *Let $C = \{F_1, \ldots, F_n\}$ be a set of formulae in NNF containing no existential quantifiers: The* clausification *of C, written as* $\mathrm{Cls}(C)$ *is the result of the following replacements* **(1)** *Replace $A \vee (B \wedge C)$ by $(A \vee B) \wedge (A \vee C)$.* **(2)** *Replace $(A \wedge B) \vee C$ by $(A \vee C) \wedge (B \vee C)$.* **(3)** *Replace $\forall x A$ by $A[x := X]$, where X is a designated variable symbol not occurring in A.* **(4)** *If one of the F_i has form $A \wedge B$, then replace F_i by A and B separately.*

We now have to show that the transformations translate formulae in $\mathcal{GF}$ into guarded clause sets. Transformation Na and NNF are unproblematic, since the result is still in $\mathcal{GF}$. There is only the small problem that in Condition 5 in Definition 1, the formula $\forall(\overline{x}a \rightarrow A)$ has to be replaced by $\forall \overline{x}(\neg\, a \vee A)$. It will turn out that Condition 6 can be completely dropped for formulae in NNF.

Theorem 4. *Let $C \in \mathcal{GF}$. Then* **(1)** $C' = (\mathrm{Na}; \mathrm{NNF})(C) \in \mathcal{GF}$. *(using the modification for the negation normal form),* **(2)** $C'' = \mathrm{Struct}_{\mathcal{GF}}(C') \in \mathcal{GF}$, **(3)** $(\mathrm{Sk}; \mathrm{Cls})(C'')$ *is a guarded clause set.*

Proof. We study the steps made in the transformation: Na and NNF can be characterized by a set of rewrite-rules, none of which introduces a free variable in a formula. Let $\Phi = \forall \overline{x}(a \rightarrow A)$ or $\Phi = \exists \overline{x}(a \wedge A)$ be a guarded quantification.

Φ will remain guarded under any rewrite step completely inside A. Similarly if A occurs in the X or Y of a rewrite rule $(X \text{ op } Y) \Rightarrow \Phi(X, Y)$ then A is copied without problems. The only possible problem is when $\forall \overline{x}(a \rightarrow A)$ rewrites to $\forall \overline{x}(\neg\, a \vee A)$, but for this case we extended the definition of the guarded fragment. Next we consider $\text{Struct}_{\mathcal{GF}}$. The formula $\forall \overline{x}\overline{y}[\neg\, a \vee \neg\, \alpha(\overline{y}) \vee A]$ is guarded, since a is a guard, and A is not affected. Any quantification in which $\forall \overline{x}(\neg\, a \vee A)$ occurs remains guarded when it is replaced by $\alpha(\overline{y})$, because no new free variables are introduced. Quantifications inside A are not affected by this operation.
For Skolemization note that every existential quantifier occurs either outside the scope of any $\forall$-quantifier, in that case it will be replaced by a constant, or in the A of a guarded formula $\forall \overline{x}(\neg\, a \vee A)$, where A does not contain any universal quantifiers. In this case the existential quantifier will be replaced by a functional term $f(\overline{x})$, where $\overline{x}$ contains exactly the set of variables occurring in the guard. The result is a formula in which all universal quantifiers are guarded, and all functions are Skolem functions. They are either constants or contain all variables of the guarded quantification in which they occur.
After that the formulae $\forall \overline{x}(\neg\, a \vee A)$ will be factored into guarded clauses

$$\forall \overline{x}(\neg\, a \vee A_1), \ldots, \forall \overline{x}(\neg\, a \vee A_n),$$

and the result is a guarded clause set. Every non-ground functional term in an A_i is obtained by Skolemization, and contains exactly the free variables of a. This ensures that the literals in A_i are weakly covering, because every variable occurring in A_i occurs in a.

Example 3. The guarded formula

$$\exists x\ n(x) \wedge \forall y[a(x,y) \rightarrow \neg\ \exists z < p(x,z) \wedge (\forall x\ a(x,z) \rightarrow (b(z,z) \wedge c(x,x))\ )\ >]$$

is translated as follows: First (Na; NNF) results in

$$\exists x\ n(x) \wedge \forall y[\neg\, a(x,y) \vee \forall z < \neg\, p(x,z) \vee (\exists x\ a(x,z) \wedge (\neg\, b(z,z) \vee \neg\, c(x,x))\ )\ >].$$

After that $\text{Struct}_{\mathcal{GF}}$ results in the following set of formulae:

$$\exists x[\ n(x) \wedge \alpha(x)], \quad \forall xy[\neg\, a(x,y) \vee \neg\, \alpha(x) \vee \beta(x)],$$

$$\forall xz[\neg\, p(x,z) \vee \neg\, \beta(x) \vee (\exists x\ a(x,z) \wedge (\neg\, b(z,z) \vee \neg\, c(x,x))\ )].$$

Then Sk results in:

$$n(c) \wedge \alpha(c), \quad \forall xy[\neg\, a(x,y) \vee \neg\, \alpha(x) \vee \beta(x)],$$

$$\forall xz[\neg\, p(x,z) \vee \neg\, \beta(x) \vee (a(f(x,z),z) \wedge (\neg\, b(z,z) \vee \neg\, c(f(x,z),f(x,z))\ ))].$$

Clausification produces:

$\{n(c)\} \quad \{\alpha(c)\} \quad \{\neg\, a(X,Y), \neg\, \alpha(X), \beta(X)\}$
$\{\neg\, p(X,Z), \neg\, \beta(X), a(f(X,Z),Z)\}$
$\{\neg\, p(X,Z), \neg\, \beta(X), \neg\, b(Z,Z), \neg\, c(f(X,Z),f(X,Z))\}$

6 The Resolution Strategy

Now that we have transformed the guarded formulae into a guarded clause set, we can define the resolution strategy. The strategy is defined by the following order. In order to prove that it is a decision procedure we have to show that the set of clauses that can be derived is finite, and that the strategy is complete.

Definition 12. *We define the following order* $\sqsubset$ *on literals:* **(1)** $A \sqsubset B$ *if* $\text{Vardepth}(A) < \text{Vardepth}(B)$, *or* **(2)** $A \sqsubset B$ *if* $\text{Var}(A) \subset \text{Var}(B)$.

Note that the cases are not disjunctive. It is easily checked that $\sqsubset$ is an order on guarded clause sets, so every clause has maximal literals. If a clause c contains non-ground functional terms, then the literals with maximal Vardepth are maximal. Otherwise at least the guard is maximal. It is always the case that every maximal literal of a clause c contains all variables of c.

Lemma 2. *Let C be a finite set of guarded clauses. Let $\overline{C}$ be the set of clauses that can be derived from C by $\sqsubset$-ordered resolution, and by unrestricted factorization. Then:* **(1)** *Every clause in $\overline{C}$ is guarded.* **(2)** $\text{Varnr}(\overline{C}) \leq \text{Varnr}(C)$. **(3)** $\text{Vardepth}(\overline{C}) \leq \text{Vardepth}(C)$.

Proof. We use induction on the derivation. A clause in $\overline{C}$ is either a clause from C, derived by resolution, or derived by factorization.
For initial clauses from C, the situation is trivial.

Let c be obtained from two guarded clauses c_1 and c_2 by resolution. We show that **(1)** c is guarded, **(2)** $\text{Varnr}(c) \leq \text{Varnr}(c_1)$ or $\text{Varnr}(c) \leq \text{Varnr}(c_2)$, and that **(3)** $\text{Vardepth}(c) \leq \text{Vardepth}(c_1)$ or $\text{Vardepth}(c) \leq \text{Vardepth}(c_2)$. Write $c_1 = \{A_1\} \cup R_1$, $c_2 = \{A_2\} \cup R_2$, where A_1 and A_2 are the complementary literals resolved upon. Let Θ be the mgu that was used. If both c_1 and c_2 are ground, then the result is immediate. If one of c_1, c_2 is ground, say c_1, then $A_2\Theta = A_1\Theta = A_1$ is ground, and $R_2\Theta$ is ground, because $\text{Var}(R_2) \subseteq \text{Var}(A_2)$. Because of this c must be ground. Then c is guarded, $\text{Vardepth}(c) = -1$, and $\text{Varnr}(c) = 0$.
If both c_1 and c_2 are non-ground, then let d be the maximum of $\text{Vardepth}(c_1)$ and $\text{Vardepth}(c_2)$. Let n be the maximum of $\text{Varnr}(c_1)$ and $\text{Varnr}(c_2)$. By Theorem 1, $\text{Vardepth}(A_1\Theta) = \text{Vardepth}(A_2\Theta) \leq d$, and $\text{Varnr}(A_1\Theta) = \text{Varnr}(A_2\Theta) \leq n$.
Since $\text{Vardepth}(R_i) \leq \text{Vardepth}(A_i)$, and $\text{Var}(R_i) \subseteq \text{Var}(A_i)$, using Lemma 1, $\text{Vardepth}(R_i\Theta) \leq \text{Vardepth}(A_i\Theta)$ and $\text{Var}(R_i\Theta) \subseteq \text{Var}(A_i\Theta)$. This together ensures that $\text{Vardepth}(R_1\Theta \cup R_2\Theta) \leq d$, and $\text{Varnr}(R_1\Theta \cup R_2\Theta) \leq n$.
It remains to show that c is guarded. Using Theorem 1 and Theorem 3 it follows that every literal in $R_i\Theta$ is weakly covering.
We still have to show that for every $B \in c$ with $\text{Vardepth}(B) > 1$, it is the case that $\text{Var}(B) = \text{Var}(c)$, and that c contains a negative literal G, s.t. $\text{Vardepth}(G) = 1$ and G contains all variables of c.

Assume without loss of generality that $\text{Vardepth}(A_1) \leq \text{Vardepth}(A_2)$. If A_2 is a guard of c_2, then A_1 is not a guard, because guards are negative. Because in that case $\text{Vardepth}(A_1) = \text{Vardepth}(A_2) = 1$, we can exchange c_1 and c_2. So we may assume that A_2 is not a a guard, and $\text{Vardepth}(A_1) \leq \text{Vardepth}(A_2)$. Let G be a guard of c_2. We have $G \in R_2$.
The mgu Θ has the property that for every variable X of c_2, the result $X\Theta$ is either ground or a variable, because otherwise Theorem 1 would be violated. It follows easily that $G\Theta$ contains all variables of $R_2\Theta$, and every literal $B\Theta \in R_2\Theta$ with $\text{Vardepth}(B\Theta) > 1$ contains all variables of $G\Theta$.
$G\Theta$ also contains all variables of $R_1\Theta$. Because $\text{Var}(R_1) \subseteq \text{Var}(A_1)$ we have $\text{Var}(R_1\Theta) \subseteq \text{Var}(A_1\Theta)$. From $\text{Var}(A_2) \subseteq \text{Var}(G)$ it follows that that $\text{Var}(A_2\Theta) = \text{Var}(A_1\Theta) \subseteq \text{Var}(G\Theta)$.
It remains to show that every literal $B\Theta$ in $R_1\Theta$, with $\text{Vardepth}(B\Theta) > 1$ contains all variables of $G\Theta$. If $\text{Vardepth}(B\Theta) > 1$, then either $\text{Vardepth}(B) > 1$, or $\text{Vardepth}(B) = 1$, and a variable in B was replaced by a non-ground, functional term. In both cases $\text{Vardepth}(A_1) > 1$ and $\text{Var}(B\Theta) = \text{Var}(A_1\Theta) = \text{Var}(A_2\Theta)$. Because of the property above of Θ it must be the case that $\text{Vardepth}(A_2) > 1$. But then $\text{Var}(A_2) = \text{Var}(G)$ and $\text{Var}(A_2\Theta) = \text{Var}(G\Theta)$.

Let c be obtained from c_1 by factorization. We show that **(1)** c is guarded, **(2)** $\text{Varnr}(c) \leq \text{Varnr}(c_1)$, and **(3)** $\text{Vardepth}(c) \leq \text{Vardepth}(c_1)$. If c is ground then the situation is trivial, otherwise we have $c_1 = \{A_1, A_2\} \cup R$, and $c = \{A_1\Theta\} \cup R\Theta$, where Θ is the mgu of A_1 and A_2.
It is sufficient to show that for every variable X of c_1, the result $X\Theta$ is either a variable or ground.
We may assume that $\text{Vardepth}(A_1) \leq \text{Vardepth}(A_2)$. By Theorem 1, $\text{Vardepth}(A_2\Theta) \leq \text{Vardepth}(A_2)$. This implies that at least for all variables in A_2 the result is a variable or ground. Now if $\text{Vardepth}(A_2) > 1$, then A_2 contains all variables of c_1, and we are ready. Otherwise $\text{Vardepth}(A_1) = \text{Vardepth}(A_2) = 1$. In that case the desired property of Θ is immediate.

It remains to show that the set of clauses is finite. For this we need:

Lemma 3. *1. Let c be a non-ground factor of clause c_1. Clause c contains no ground terms, which are not in c_1.*
2. Let c be a non-ground resolvent of clauses c_1 and c_2. Then c contains no ground terms, that are not in c_1 or c_2.
3. Let c be a resolvent of c_1 and c_2, where c_1 is ground, and c_2 is non-ground. Then $\text{Depth}(c) \leq \text{Depth}(c_1)$ *or* $\text{Depth}(c) \leq \text{Depth}(c_2)$.
4. Let c be a resolvent of c_1 and c_2, which are both ground. Then $\text{Depth}(c) \leq \text{Depth}(c_1)$, *or* $\text{Depth}(c) \leq \text{Depth}(c_2)$.

Part (1) and (2) follow from Theorem 2. Part (3) and (4) are easily checked.

Lemma 4. *Let C be a finite set of guarded clauses. Let $\overline{C}$ be its closure under $\sqsubset$-ordered resolution, and factoring (unrestricted). Then $\overline{C}$ is finite in size.*

Proof. The difficulty is that, although $\text{Vardepth}(\overline{C}) \leq \text{Vardepth}(C)$, and $\text{Varnr}(\overline{C}) \leq \text{Varnr}(C)$, we have no upper bound for the ground terms.
Let C_{ng} be the set of non-ground clauses in C. Let $\overline{C}_{ng}$ be the set of non-ground clauses that can be derived from C_{ng}. (So, $\overline{C}_{ng}$ is the total set of non-ground clauses that can be derived)
It follows from Lemma 3 that $\overline{C}_{ng}$ does not contain a ground term that is not in C_{ng}. Hence $\overline{C}_{ng}$ is finite in size.
After that $\overline{C}$ can be obtained from $\overline{C}_{ng}$ by deriving only ground clauses. It follows from Lemma 3, that $\overline{C}$ is finite in size.

It remains to show the completeness. The order is non-liftable, i.e. does not satisfy $A \sqsubset B \Rightarrow A\Theta \sqsubset B\Theta$, for example we have:

1. $p(s(0), X) \sqsubset p(0, s(X))$ and $p(X, s(0)) \sqsubset p(s(X), 0)$. The substitution $\{X := 0\}$ results in a conflict.
2. Also $\neg\, p(X, X) \sqsubset \neg\, q(X, Y)$ and $\neg\, q(X, X) \sqsubset \neg\, p(X, Y)$. The substitution $\{X := Y\}$ results in a conflict.

The completeness proof is based on the resolution game ([Nivelle94], or [Nivelle95]). We need some technical preparation: A literal A is *normal* if variable X_{i+1} occurs only after an occurrence of variable X_i. (When the literal is written in the standard notation. We assume a fixed enumeration of the variables). We write $\overline{A}$ for the normalization of A. Every literal A can be renamed into exactly one normal literal, called the *normalization* of A. If two literals are renamings of each other, they have the same normalization.

Definition 13. *Let $\Theta = \{V_1 := t_1, \ldots, V_n := t_n\}$ be a substitution. The complexity of Θ, written as $\#\Theta$ equals $\#t_1 + \cdots + \#t_n$.*

Theorem 5. *Resolution, using $\sqsubset$ is complete for guarded clause sets.*

Proof. Let C be an unsatisfiable guarded clause set. Let $\overline{C}$ be the set of clauses that can be obtained from C using $\sqsubset$-ordered resolution, and $\sqsubset$-ordered factoring. We show that $\overline{C}$ must contain the empty clause.
Write $C = \{c_1, \ldots, c_n\}$. Let $\Theta_{1,1}, \ldots, \Theta_{1,l_1}, \ldots, \Theta_{n,1}, \ldots, \Theta_{n,l_n}$ be a list of substitutions such that the set of clauses $c_1\Theta_{1,1}, \ldots, c_1\Theta_{1,l_1}, \ldots, c_n\Theta_{n,1}, \ldots, c_n\Theta_{n,l_n}$ is propositionally unsatisfiable. We have each $l_i \geq 0$. We call this clause set the Herbrand set.
First we annotate each clause in the Herbrand set with its representing clauses as follows: For each $c_i = \{A_1, \ldots, A_p\}$ and substitution $\Theta_{i,j}$, the set C_{hb} contains the clause

$$\{A_1\Theta_{i,j}{:}A_1, \ldots, A_p\Theta_{i,j}{:}A_p\}.$$

The objects of the form $a{:}A$ are called *indexed literals*. Extend the order $\sqsubset$ to indexed literals by $(a{:}A) \sqsubset (b{:}B)$ iff $A \sqsubset B$. Then define the following resolution and factoring rule for indexed clause sets:

resolution From $\{a{:}\,A_1\} \cup R_1$ and $\{\neg\ a{:}\,A_2\} \cup R_2$ derive $R_1\Theta \cup R_2\Theta$.
factoring From $\{a{:}\,A_1, a{:}\,A_2\} \cup R$ derive $\{a{:}\,A_1\Theta\} \cup R\Theta$.

In both cases Θ is the mgu. The result of Θ on an indexed literal $b{:}\,B$ is defined as $b{:}\,(B\Theta)$. The literals resolved upon, and one of the literals factored upon must be maximal.
Given this resolution and factoring rule, let $\overline{C}_{hb}$ be the closure of C_{hb}. It is clear that if we can prove that $\overline{C}_{hb}$ contains the empty clause, then $\overline{C}$ contains the empty clause.
In order to do this define the following resolution game $\mathcal{G} = (P, \mathcal{A}, \prec)$, and initial clause set $C_\mathcal{G}$:

- The set P of literals equals the set of literals that occur in the Herbrand set.
- The initial indexed clause set $C_\mathcal{G}$ consists of the following indexed clauses: For each indexed clause $\{a_1{:}\,A_1, \ldots, a_p{:}\,A_p\}$ in C_{hb}, there is the following clause in $C_\mathcal{G}$:
$$\{a_1{:}\,(k, \overline{A}_1), \ldots, a_p{:}\,(k, \overline{A}_p)\}.$$
Here $k = \#\Theta$, where Θ is the substitution that makes $a_i = A_i\Theta$. The $\overline{A}_1, \ldots, \overline{A}_p$ are the normalizations of the $A_1, \ldots, A_p$.
- The set $\overline{C}_\mathcal{G}$ is defined as $C_\mathcal{G}$, but taking $\overline{C}_{hb}$ as a starting point, instead of C_{hb}.
- The set $\mathcal{A}$ of attributes is obtained from $\overline{C}_\mathcal{G}$ as the set of $(k, \overline{A})$, for which there is an indexed literal $a{:}\,(k, \overline{A})$ in $\overline{C}_\mathcal{G}$.
- The order $\prec$ is defined from: $a_1{:}\,(i_1, C_1) \prec a_2{:}\,(i_2, C_2)$ if **(1)** $i_1 < i_2$, or **(2)** $i_1 = i_2$ and $(\mathrm{Varnr}(C_1) < \mathrm{Varnr}(C_2)$ or $\mathrm{Vardepth}(C_1) < \mathrm{Vardepth}(C_2))$.

This completes the resolution game.
If we can show that $\overline{C}_\mathcal{G}$ contains the empty clause then we are done, since this implies that $\overline{C}_{hb}$ contains the empty clause.
We show that $\overline{C}_\mathcal{G}$ contains the empty clause by showing that it is a saturation of $C_\mathcal{G}$, based on $\mathcal{G}$.
Let $c_1 = \{a{:}\,(k_1, \overline{A}_1)\} \cup R_1$ and $c_2 = \{\neg\ a{:}\,(k_2, \overline{A}_2)\} \cup R_2$ be clauses in $\overline{C}_\mathcal{G}$, for which $a{:}\,(k_1, \overline{A}_1)\}$ and $\neg\ a{:}\,(k_2, \overline{A}_2)$ are maximal. Let $d_1 = \{a{:}\,A_1\} \cup S_1$ and $d_2 = \{\neg\ a{:}\,A_2\} \cup S_2$ be the clauses in $\overline{C}_{hb}$ from which d_1 and d_2 originate. Then $a{:}\,A_1$ and $\neg\ a{:}\,A_2$ must be maximal in d_1 and d_2. For if some literal in d_1 would be larger, the corresponding literal in c_1 would also be larger, in c_1, since all the k in the (k, B) of the indices are equal. The same is true for d_2. Because of this d_1 and d_2 have a resolvent d. We must show that the clause $c \in \overline{C}_\mathcal{G}$, resulting from d is a reduction of the resolvent of c_1 and c_2.
If one of the literals $b{:}\,B$ from R_1 is replaced by $b{:}\,B\Sigma$, (where Σ is the mgu), then $b{:}\,(k, B)$ can be replaced by $b{:}\,(k', B\Sigma)$, where $k' < k$. The same is true for the literals from R_2. This ensures that the result is a reduction. The situation in the case of factoring is analogous.

The order $\sqsubset$ as we have defined it here is very basic, and it could be refined further to improve the efficiency. Every order $\sqsubset' \supseteq \sqsubset$ can be used to decide the guarded fragment.

7 Conclusions & Further Work

We have shown that it is possible to effectively decide the guarded fragment by resolution. The proof that the resolution refinement is complete and terminating could be used as proof for the decidability of this fragment, but they offer more than that. They also define practical decision procedures, using refinements that are standard to the theorem proving community.
Future work should be the comparison of the complexity of the procedures with the theoretical complexity results obtained in [Graedel97]. Also some solution should be found for transitivity axioms. Transitivity axioms are non-guarded, and it has been shown in [Graedel97] that adding transitivity axioms leads to undecidability. Nevertheless there are modal logics ($S4$, and $K4$) based on transitive frames, that are decidable. So it must be possible to combine some weaker version of the guarded fragment with transitivity. Another point to look at is back translation. As some people prefer to see proofs in modal logic, rather than proofs in first order logic, it is useful to look into possiblities of translating the proofs in the guarded fragment back to proofs in the modal logics.

References

[ANB96] H. Andréka, J. van Benthem, I. Németi, Modal Languages and Bounded Fragments of Predicate Logic, ILLC Research Report ML-96-03, 1996.

[BFL94] M. Baaz, C. Fermüller, A. Leitsch, A Non-Elementary Speed Up in Proof Length by Structural Clause Form Transformation, In LICS 94.

[BL94] M. Baaz, A. Leitsch, On Skolemization and Proof Complexity, Fundamenta Informatica, Vol. 20-4, 1994.

[Benthem96] J. van Benthem, Exploring Logical Dynamics, CSLI Publications, Stanford, California USA, 1996.

[Benthem97] J. van Benthem, Dynamic Bits and Pieces, LP-97-01, Research Report of the Institute for Logic, Language and Information, 1997.

[BGG96] E. Börger, E. Grädel, Y. Gurevich, The Classical Decision Problem, Springer Verlag, Berlin Heidelberg, 1996.

[Catach91] L. Catach, TABLEAUX, a general theorem prover for modal logics, Journal of automated reasoning 7, pp. 489-510, 1991.

[CL73] C-L. Chang, R. C-T. Lee, Symbolic Logic and Mechanical Theorem Proving, Academic Press, New York, 1973.

[DG79] B. Dreben, W.D. Goldfarb, The Decision Problem, Solvable Classes of Quantificational Formulas, Addision-Wesley Publishing Company, Inc. 1979.

[EnjFar89] P. Enjalbert, L. Fariñas del Cerro, Modal resolution in clausal form, Theoretical Computer Science 65, 1989.

[FarHerz88] L. Fariñas del Cerro and A. Herzig, linear modal deductions, CADE '88, pp. 487-499, 1988.

[FLTZ93] C. Fermüller, A. Leitsch, T. Tammet, N. Zamov, Resolution Methods for the Decision Problem, Lecture Notes in Artificial Intelligence 679, Springer Verlag, 1993.

[Fitting88] M. Fitting, First-order modal tableaux, Journal of automated resoning 4, pp. 191-213, 1991.

[Fitting91] M. Fitting, Destructive Modal Resolution, Journal of Logic and Computation, volume 1, pp. 83-97, 1990.

[Foret92] A. Foret, Rewrite rule systems for modal propositional logic, Journal of logic programming 12, pp. 281-298, 1992.

[Graedel97] E. Grädel, On the Restraining Power of Guards, manuscript, 1997.

[Joyner76] W. H. Joyner, Resolution Strategies as Decision Procedures, J. ACM 23, 1 (July 1976),l pp. 398-417, 1976.

[Ladner77] R.E. Ladner, The computational complexity of provability in systems of modal propositional logic, SIAM Journal on Computing 6, pp 467-480, 1977.

[McCune95] W. W. McCune, Otter 3.0 Reference Manual and Guide, Argonne National Laboratory, Mathematics and Computer Science Division, can be obtained from **ftp.mcs.anl.gov**, directory **pub/Otter**, 1995.

[Nivelle92] H. de Nivelle, Generic modal resolution, Technical Report 92-90, Delft University of Technology, fac. TWI, 1992.

[Nivelle93] Generic Resolution in Propositional Modal Systems, in LPAR 93, Springer Verlag Berlin, 1993.

[Nivelle94] Resolution Games and Non-Liftable Resolution Orderings, in CSL 94, pp. 279-293, Springer Verlag, 1994.

[Nivelle95] H. de Nivelle, Ordering Refinements of Resolution, Ph. D. Thesis, Delft University of Technology, 1995.

[Nivelle98] H. de Nivelle, Resolution Decides the Guarded Fragment, ILLC-Report CT-1998-01, 1998.

[Ohlbach88a] H.J. Ohlbach, A resolution calculus for modal logics, PhD thesis, Universität Kaiserslautern, 1988.

[Ohlbach88b] H.J. Ohlbach, A resolution calculus for modal logics, CADE '88, pp. 500-516, 1988.

[OWbach95] H-J. Ohlbach, C. Weidenbach, A note on Assumptions About Skolem Functions, Journal of Automated Reasoning 15, Vol. 2, pp. 267-275, 1995.

[Robinson65] J. A. Robinson, A Machine Oriented Logic Based on the Resolution Principle, Journal of the ACM, Vol. 12. pp. 23-41, 1965

[ATP97] The CADE-13 Automated Theorem Proving System Competition, Journal of Automated Reasoning Special Issue, Vol. 18, No. 2, Edited by G. Sutcliffe and C. Suttner, 1996.

[Tammet90] T. Tammet, The Resolution Program, Able to Decide some Solvable Classes, in COLOG-88, Springer LNCS, pp. 300-312, 1990.

[Wbach96] C. Weidenbach, (Max-Planck-Institut für Informatik), The Spass & Flotter Users Guide, Version 0.55, can be obtained from **ftp.mpi-sb.mpg.de**, directory **pub/SPASS**, 1997.

[Zamov72] N.K. Zamov, On a Bound for the Complexity of Terms in the Resolution Method, Trudy Mat. Inst. Steklov 128, pp. 5-13, 1972.