Geometric Resolution: A Proof Procedure based on Finite Model Search

Hans de Nivelle and Jia Meng

Max-Planck Institut für Informatik, Germany, nivelle@mpi-inf.mpg.de
National ICT Australia, Australia, Jia.Meng@nicta.com.au

Abstract. We present a proof procedure that is complete for first-order logic, but which can also be used when searching for finite models. The procedure uses a normal form which is based on geometric formulas. For this reason we call the procedure *geometric resolution*. We expect that the procedure can be used as an efficient proof search procedure for first-order logic. In addition, the procedure can be implemented in such a way that it is complete for finding finite models.

1 Introduction

For many applications of automated theorem proving, knowing a counter model to a wrong conjecture is as useful as knowing that a conjecture is true. When a theorem prover fails to find a proof, the user might try to give more resources to the prover, tune the settings of the prover, or try to get another theorem prover. When the prover returns a model, the user will concentrate on trying to correct the conjecture.

Variants of resolution [11, 1] are quite succesful in finding proofs. Although resolution can be modified into decision procedures for many decidable fragments [6–8], there is no general way of extracting models from these procedures. Also there exists no known general method for making resolution complete in finding finite models.

The model evolution calculus of [2] is complete and is guaranteed to terminate on clause sets without function symbols. However, because the model evolution calculus operates on Skolemized formulas, this not include formulas with existential quantifiers in the scope of universal quantifiers. In contrast, our proof procedure can be implemented in such a way that it terminates on all formulas that have a finite model.

There exist various approaches to searching for finite models: One approach is based on guessing a possible size of the domain, instantiating the clauses within the domain, and applying propositional reasoning on the result. If the result is satisfiable, then a model has been found. Otherwise, the size of the domain has to be increased. Approaches among this line are the systems MACE [10] and ModGen [9]. Alternatively, one can search for a model by direct search. This approach is followed for example in [12], [13] and [14].

Our algorithm is somewhat related to this last approach but it has also elements from other approaches. It consists of a search algorithm which is almost identical to the algorithm in [5], but enhanced with lemma generation. In [5], the algorithm searches for a model by exhaustive search. When a disjunction is encountered, all members of the disjunction are tried. When an existential quantifier is encountered, first all existing elements in the interpretation are tried. If this fails, a new element is tried.
This naive model search algorithm is refutationally complete, and also guaranteed to find finite models when breadth-first search is used, but very inefficient. The major source of inefficiency are the existential quantifiers. Every time, when a subformula of form $\exists y\ P(\overline{d}, y)$ needs to be made true, all elements in the interpretation have to be tried as candidates for y.
In order to reduce the search complexity, we add lemma learning to the procedure. We introduce a calculus with which it is always possible, at each choice point, after all subbranches have been refuted, to derive a new formula which closes the search attempt before the choice point. Because the derived lemmas contain variables, the lemmas give a most general reason why the branch has been closed. In this way, repetition of identitical work can be largely reduced. In particular, we will show that in many cases enumeration of domain elements can be avoided.
The way we use lemma generation is related to lemma generation in DPLL [15]. The main difference with lemma generation in DPLL is that our calculus operates on formulas with variables. In fact, one could say that our calculus for lemma generation relates to the naive model search algorithm in the same way as resolution with superposition relates to the model construction procedure of [1]. The essential difference is that we actually run the naive search algorithm and use the lemma calculus to improve its efficiency. For superposition, the model construction procedure is used only as a tool for proving completeness. Our lemma calculus could also be used in this way (i.e. as a saturation-based proof search method), but one would loose the ability to find finite models.

2 Geometric Formulas

The model search algorithm and the lemma calculus operate on what we call *geometric formulas*. The formulas are related to the formulas used in [4], but not exactly the same. In [4], the right hand side of a formula is a disjunction of existentially quantified conjunctions. Here we simplified the right hand sides by allowing only one operator at the same time.

Geometric formulas play a role similar to clauses in saturation-based theorem proving. The main difference is that geometric formulas do not contain constant and function symbols. Instead they may contain existential quantifiers. We show in Section 3 that every first-order formula (possibly containing equality) can be transformed to a set of geometric formulas, so that no generality is lost.

Definition 1. *We assume an infinite set of* variables $\mathcal{V}$. *A* variable atom *is defined by one of the following two forms:*

– $x_1 \not\approx x_2$, *with* $x_1, x_2 \in \mathcal{V}$ *and* $x_1 \neq x_2$.
– $p(x_1, \ldots, x_n)$ *with* $n \geq 0$ *and the* $x_i \in \mathcal{V}$.

There are no constants and no function symbols in variable atoms. There are no positive equalities. In principle, one could allow disequalities of form $v \not\approx v$, but since these are known to be false, there is no need to keep such equalities. Geometric formulas are built from variable atoms. We give the definition:

Definition 2. *A* geometric formula *has form*

$$\forall \overline{x}\ A_1(\overline{x}) \wedge \cdots \wedge A_p(\overline{x}) \wedge x_1 \not\approx x'_1 \wedge \cdots \wedge x_q \not\approx x'_q \rightarrow Z(\overline{x}),$$

in which $p \geq 0$, $q \geq 0$, *and the* $x_1, x'_1, \ldots, x_q, x'_q \in \overline{x} \subseteq \mathcal{V}$.
The right hand side $Z(\overline{x})$ *must have one of the following three forms:*

1. *The false constant* $\bot$.
2. *A non-empty disjunction of atoms* $B_1(\overline{x}) \vee \cdots \vee B_r(\overline{x})$ *with* $r > 0$.
3. *An existential formula of form* $\exists y\ B(\overline{x}, y)$ *with* $y \in \mathcal{V}$ *but* $y \notin \overline{x}$. *The variable* y *must occur in* $B(\overline{x}, y)$.

A formula of the first type is called lemma. *A formula of the second type is called* disjunctive. *A formula of the third type is called* existential.

Our notations can be clarified as follows:

– In $\forall \overline{x}$, $\overline{x}$ denotes an enumeration of $\overline{x}$, in arbitrary order, mentioning each variable of $\overline{x}$ exactly once. The scope of $\forall \overline{x}$ is the complete geometric formula.
– In $A(\overline{x})$, $\overline{x}$ denotes a sequence of variables from $\overline{x}$. Variables may be repeated, and variables may be omitted.
– Latin letters A, B denote variables atoms that are not disequalities.
– In later sections, we will use expressions of form $\Phi(\overline{x})$ or $\phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x})$ for conjunctions of variable atoms of both types with variables in $\overline{x}$.

Note that in each of the three cases, the free variables of $Z(\overline{x})$ are within $\overline{x}$. However, since $A_1(\overline{x}) \wedge \cdots \wedge A_p(\overline{x})$ need not contain all variables of $\overline{x}$, this does not imply that geometric formulas are range restricted. (An implication is range restricted if each variable occurring in the right hand side also occurs in the left hand side) Types 1 and 2 could be merged if we would allow $r = 0$ in type 2. We prefer to keep the types distinct, because the roles of the formulas will be different in the calculus.

Example 1. Propositional clauses can be replaced by geometric formulas of type 1 or type 2 with empty $\overline{x}$. For example, $A \vee B \vee \neg C$ can be replaced by $C \rightarrow A \vee B$. The clause $A \vee B$ is already a geometric formula. The clause $\neg A \vee \neg B$ can be replaced by $A \wedge B \rightarrow \bot$.

Example 2. Equalities can be translated into geometric formulas. Consider the positive equality $f(a) \approx f(b)$. First one introduces relations A, B, F and geometric formulas: $\exists y\ A(y)$, $\exists y\ B(y)$, $\forall x\ \exists y\ F(x, y)$. Using those, the equality can be

expressed by $\forall x_1, x_2, x_3, x_4\ A(x_1) \wedge F(x_1, x_2) \wedge B(x_3) \wedge F(x_3, x_4) \wedge x_2 \not\approx x_4 \rightarrow \bot$.

The negative equality $f(a) \not\approx f(b)$ can be expressed by the geometric formula $\forall x_1, x_2, x_3\ A(x_1) \wedge S(x_1, x_2) \wedge B(x_3) \wedge S(x_3, x_2) \rightarrow \bot$.

3 Conversion from FOL to Geometric Logic

The conversion from first-order logic to geometric logic is analogous to the clause transformation that is used for resolution. The main difference is that functions are replaced by existential quantifiers, instead of replacing existential quantifiers by functions.

Theorem 1. *There exists a transformation from first-order formulas to finite sets of geometric formulas, which can be efficiently computed. For each non-empty set D holds: The formula F has a model D as domain iff the translation $G_1, \ldots, G_m$ (read as conjunction) has a model with D as domain.*

It is possible to obtain a polynomial transformation, but we prefer not to stress this, because a polynomial transformation need not be the best in practical cases. We stress the domain of the interpretations (instead of simply stating equisatisfiability) because we are interested both in models and in proofs.

As a starting point of the transformation, we assume that F is a formula in negation normal form. Because geometric formulas do not allow function symbols, we will replace functions by serial relations, somewhat similar to the transformation in [9]. The difference is that here we use only seriality axioms, while there also functionality axioms are needed.

When removing functions, constants are treated as 0-arity functions, these will be also deleted. We call the removal operation *anti Skolemization*. In the transformation, we assume that F is standardized apart. (A formula is standardized apart if there are no two quantifiers that quantify the same variable)

Definition 3. *Let F be a formula in negation normal form. We assume that for each function symbol f occurring in F with arity a, there exists a unique predicate symbol P_f with arity $(a+1)$. We define S_f as the seriality axiom for P_f,*

$$\forall x_1, \ldots, x_a\ \exists y\ P_f(x_1, \ldots, x_a, y).$$

We define a replacement sequence $F_1, F_2, \ldots, F_n$ which starts with $F_1 = F$, and which eliminates occurrences of function symbols one-by-one.

Let $f(x_1, \ldots, x_a)$ be a term which occurs in F_i, and in which the $x_1, \ldots, x_a$ are variables. (not necessarily distinct)
Let A be the smallest subformula of F_i containing all occurrences of $f(x_1, \ldots, x_a)$.
Write F_i in the form $F_i[\ A[\ f(x_1, \ldots, x_a)\ ]\ ]$.
Let P_f be the predicate symbol assigned to f. Let α be a variable that does not occur in F_i. Then F_{i+1} is defined as

$$F_i[\ \forall \alpha\ (\neg P_f(x_1, \ldots, x_a, \alpha) \vee A[\alpha]\ )\ ].$$

At some point, the sequence will reach a formula F_n which has no remaining function symbols. Then the anti-Skolemization *of F equals*

$$S_{f_1} \wedge \cdots \wedge S_{f_q} \wedge F_n.$$

Theorem 2. *We use the term* D-model *for a* model with D as domain.

Let F be a formula. Let F_n be its anti-Skolemization. Let D be a non-empty set. Then F has a D-model iff F_n has a D-model.

Proof. Let the seriality axioms S_f and the sequence $F_1, F_1, \ldots, F_n$ with $F = F_1$ be defined as in Definition 3.

First assume that F_1 has a D-model. We need to show that $S_{f_1} \wedge \cdots \wedge S_{f_q} \wedge F_n$ has a D-model. In order to do this, one can interpret each predicate $P_f(x_1, \ldots, x_a, y)$ as $f(x_1, \ldots, x_a) \approx y$. Then the seriality axioms $S_{f_1}, \ldots, S_{f_q}$ are provable. For each i with $1 \leq i < n$, we have $F_i \leftrightarrow F_{i+1}$, because $A[\, f(x_1, \ldots, x_n)\,]$ is equivalent to $(\, \forall\alpha\ f(x_1, \ldots, x_n) \approx \alpha \rightarrow A[\alpha]\,)$. Iterating $n-1$ times, it follows that $S_{f_1} \wedge \cdots \wedge S_{f_q} \wedge F_n$ has a D-model.
Now assume that $S_{f_1} \wedge \cdots \wedge S_{f_q} \wedge F_n$ has a D-model. We first Skolemize the formulas of S_f. Because F_n contains no function symbols, we may assume without losing generality, that the Skolem function for S_f is f. Write Sk_f for the formula $\forall x_1, \ldots, x_a\ P_f(x_1, \ldots, x_a, f(x_1, \ldots, x_a))$.
From the soundness of Skolemization, it follows that $Sk_{f_1} \wedge \cdots \wedge Sk_{f_q} \wedge F_n$ has a D-model. Assume that the term being replaced at stage i is $f(x_1, \ldots, x_a)$. It is easily seen (using resolution) that

$$Sk_f, \forall\alpha\ (\neg P_f(x_1, \ldots, x_a, \alpha) \vee A[\alpha]\ ) \vdash A[\ [f(x_1, \ldots, x_a)\ ].$$

Iterating $n-1$ times, it follows that $Sk_{f_1} \wedge \cdots \wedge Sk_{f_q} \wedge F_0$ has a D-model.

Definition 4. *A formula is in CUDEN normal form if the operator path to each atom has form $\wedge^*\forall^*\vee^*\exists^?\neg^?$. (CUDEN stands for Conjunction, Universal quantifier, Disjunction, Existential quantifier, Negation.)*

Theorem 3. *The following replacements transform a formula from NNF to CUDEN normal form:*

1. $\forall x(A \wedge B) \Rightarrow (\forall x\ A) \wedge (\forall x\ B)$,
   $(A \wedge B) \vee C \Rightarrow (A \vee C) \wedge (B \vee C)$, $\quad (\forall x\ A) \vee B \Rightarrow \forall x\ (A \vee B)$,
   $A \vee (B \wedge C) \Rightarrow (A \vee B) \wedge (A \vee C)$, $\quad A \vee (\forall x\ B) \Rightarrow \forall x\ (A \vee B)$.
2. *If F contains a subformula $\exists y\ A(x_1, \ldots, x_a, y)$ in which $A(x_1, \ldots, x_a, y)$ is either an equality, a disequality, or not an atom, then let p be a new predicate symbol with arity $a+1$ Replace $F[\ \exists y\ A(x_1, \ldots, x_a, y)\ ]$ by*

$$F[\ \exists y\ p(x_1, \ldots, x_a, y)\ ] \wedge \forall x_1, \ldots, x_a, y\ (\ \neg p(x_1, \ldots, x_a, y) \vee A(x_1, \ldots, x_a, y)\ ).$$

Finally, a function-free formula in CUDEN normal form can be transformed into a set of geometric formulas by the following replacements:

1. Delete negative equalities from disjunctions as follows: Replace
   $\forall x_1, \ldots, x_p\ (\ \neg(x \approx x') \vee A_1 \vee \cdots \vee A_q\ )$ by
   $\forall x_1, \ldots, x_p\ (\ A_1[x := x'] \vee \cdots \vee A_q[x := x']\ )$.
2. Replace positive equalities $x \approx x'$ by negative disequalities $\neg(x \not\approx x')$.
3. If there is a disjunction of form $\forall x_1, \ldots, x_p\ A_1 \vee \cdots \vee A_q$ in which one of the A_i has form $\exists y\ B(x'_1, \ldots, x'_r, y)$, and another A_j is a positive non-equality atom, then replace $\exists y\ B(x'_1, \ldots, x'_r, y)$ by $q(x'_1, \ldots, x'_r)$ and add a new disjunction

$$\forall x'_1, \ldots, x'_r\ \neg q(x'_1, \ldots, x'_r) \vee \exists y\ B(x'_1, \ldots, x'_r, y).$$

4 Model Search without Lemma Generation

We present the model search algorithm without lemma generation. It is closely related to the methods in [4] and [5]. The main difference is that our method is able to handle equality. The algorithm with lemma generation will be presented in Section 5.

Definition 5. *We assume an infinite set of elements $\mathcal{E}$. A* ground atom *is an object of form $p(e_1, \ldots, e_a)$, with $a \geq 0$ and $e_1, \ldots, e_a \in \mathcal{E}$.*
If for some $E \subseteq \mathcal{E}$, all $e_1, \ldots, e_n$ are in E, then we call $p(e_1, \ldots, e_n)$ a ground atom over E.

In contrast to variable atoms, ground atoms have no disequalities. Ground disequalities $e \not\approx e'$ can always be evaluated.

Definition 6. *An* interpretation *is a pair (E, M) in which $E \subseteq \mathcal{E}$ is a set of elements, and M is a set of ground atoms over E.*

We will $+$ both for insertion of elements to E, and for insertion of atoms to M. If we want to add an element e to a set of elements E, we can write $E + e$. If $E = \{e_1, e_2, e_3\}$ we can write $E = e_1 + e_2 + e_3$. Similarly, if we add an atom $p(e_1, e_2)$ to M, we can write $M + p(e_1, e_2)$. A complete interpretation can be written as $(\ 0 + 1,\ \ p(0) + q(1) + p(0, 1)\ )$.

Definition 7. *A* ground substitution *Θ is a substitution from $\mathcal{V}$ to $\mathcal{E}$. We write $A(\overline{x})\Theta$ or $A(\overline{x}\Theta)$ for the* application *of Θ on $A(\overline{x})$. Let (E, M) be an interpretation, let Θ be a ground substitution:*

1. *If Θ is defined for $x_1, \ldots, x_a$, then Θ makes a variable atom of form $p(x_1, \ldots, x_a)$ true in (E, M) if $p(x_1, \ldots, x_a)\Theta \in M$.*
2. *If Θ is defined for x, x', then Θ makes the disequality $(x \not\approx x')$ true if $x\Theta \neq x'\Theta$.*

If Θ is defined for all variables in $\overline{x}$, then Θ makes a conjunction $\phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x})$ of variable atoms true in (E, M) if Θ makes all $\phi_i(\overline{x})$ true in (E, M).

Definition 8. *Let $r = \ \forall \overline{x}\ \Phi(\overline{x}) \rightarrow Z(\overline{x})$ be a geometric formula. Let (E, M) be an intepretation. We say that (E, M) makes r true if every ground substitution Θ that is defined for $\overline{x}$, either does not make $\Phi(\overline{x})$ true in (E, M), or*

1. *$Z(\overline{x})$ has form $B_1(\overline{x}) \vee \cdots \vee B_q(\overline{x})$ and Θ makes one of the $B_j(\overline{x})$ true in (E, M).*
2. *$Z(\overline{x})$ has form $\exists y\ B(\overline{x}, y)$ and there exists an element $e \in E$ such that the ground substitution $\Theta + (y := e)$ makes $B(\overline{x}, y)$ true in (E, M).*

Definition 9. *Let $r = \forall \overline{x}\ \phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x}) \rightarrow Z(\overline{x})$ be a geometric rule. Let (E, M) be an interpretation, and let Θ be a substitution which is defined on $\overline{x}$. If Θ makes $\phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x})$ true in (E, M), then we call the rule r* applicable on (E, M) with substitution Θ, *if one the following conditions is met, depending on the type of r :*

1. *$Z(\overline{x}) = \bot$. In this case we call r a* closing lemma *of (E, M).*
2. *$Z(\overline{x})$ has form $B_1(\overline{x}) \vee \cdots \vee B_q(\overline{x})$ and Θ makes none of the $B_j(\overline{x})$ true in (E, M).*
3. *$Z(\overline{x})$ has form $\exists y\ B(\overline{x}, y)$ and there exists no $e \in E$, for which $\Theta + (y := e)$ makes $B(\overline{x}, y)$ true in (E, M).*

Lemma 1. *An interpretation (E, M) makes a set of rules G true if and only if there there is no applicable rule in G.*

We present the model search algorithm. It starts with an empty model. At each stage it looks for a rule r that is applicable with some substitution Θ. If there is no applicable rule, then the interpretation makes all rules true. Otherwise, the interpretation is extended in such a way that r is not applicable anymore with Θ. In case the interpretation can be extended in more than one possible way, the algorithm has to attempt all possibilities. When applicable rules are explored in a fair fashion, the search algorithm is refutationally complete.

Definition 10. *Let G be a set of geometric formulas. $S_t(G)$ is initially defined as $S_t(G, \emptyset, \emptyset)$.*
$S_t(G, E, M)$ returns either $\bot$ or an interpretation (E', M'), s.t. $E \subseteq E'$, $M \subseteq M'$, and (E', M') makes all rules in G true. $S_t(G, E, M)$ is defined by the following cases:

1. *If G contains a lemma $\forall \overline{x}\ \phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x}) \rightarrow \bot$, which is applicable on (E, M) with ground substitution Θ, then $S_t(G, E, M)$ returns $\bot$.*
2. *If G contains a disjunctive rule $\forall \overline{x}\ \phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x}) \rightarrow B_1(\overline{x}) \vee \cdots \vee B_q(\overline{x})$ which is applicable on (E, M) with ground substitution Θ, then compute, for each j with $1 \leq j \leq q$, $\lambda_j := S_t(G, E, M + B_j(\overline{x})\Theta)$. If $\lambda_1 = \cdots = \lambda_q = \bot$, then $S_t(G, E, M)$ returns $\bot$. Otherwise $S_t(G, E, M)$ returns one of the λ_j which is different from $\bot$.*
3. *If G contains an existential rule $\forall \overline{x}\ \phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x}) \rightarrow \exists y\ B(\overline{x}, y)$, which is applicable in (E, M) with ground substitution Θ, then define, for each $e \in E$, $\Theta_e = \Theta + (y := e)$, and recursively compute $\lambda_e := S_t(G, E, M + B(\overline{x}, y)\Theta_e)$. In addition, define $\Theta_{\hat{e}} = \Theta + (y := \hat{e})$ for some $\hat{e} \notin E$, and compute $\lambda_{\hat{e}} := S_t(G, E + \hat{e}, M + B(\overline{x}, y)\Theta_{\hat{e}})$.*
   If, for each $e \in E$, $\lambda_e = \bot$, and also $\lambda_{\hat{e}} = \bot$, then $S_t(G, E, M)$ returns $\bot$. Otherwise, $S_t(G, E, M)$ returns one of the λ_e or $\lambda_{\hat{e}}$ which is different from $\bot$.

4. If there exists no applicable rule, then $S_t(G, E, M)$ returns (E, M).

It is easily seen that the algorithm $S_t(G)$ is complete, as long as applicable rules are expanded in a fair way. Due to the way existential quantifiers are treated, $S_t(G)$ also has a reasonable chance of finding finite models, but it is not guaranteed to find a finite model in case there exists one. As an example consider the set of formulas $A \vee B$, $\forall x\ A \rightarrow \exists y\ P(x, y)$, $\forall xyz\ P(x, y) \wedge P(y, z) \rightarrow P(x, z)$, $\forall x\ P(x, x) \rightarrow \bot$. The interpretation $(\emptyset, B)$ makes all rules true. However, $S_t(G)$ will probably first attempt A in the first disjunction, after which it has to construct an infinite P-chain. Completeness for finite models can be obtained by using breadth-first search, but this would make search for refutations less efficient. Although complete in theory, $S_t(G)$ is of course inefficient.

5 The Algorithm with Lemma Generation

We modify the algorithm of the previous section in such a way that it is able to learn in the following way: Whenever it cannot extend (E, M) to a model of G, the improved algorithm will not simply return $\bot$, but a lemma that closes (E, M). The lemma ensures that, whenever the algorithm enters a similar situation again, it will not search another time for a refutation, but instead reuse the lemma. It will be also possible to avoid enumerating the complete domain for an existential quantifier, in case the lemma does not depend on the actual witness chosen. Another advantage of learning is that the algorithm will be able to output proofs which can be verified in a formal calculus.

Definition 11. *A* variable substitution Σ *is a substitution from* $\mathcal{V}$ *to* $\mathcal{V}$. Most general unifiers *between non-disequality variable atoms are defined as usual.*

Note that, because we consider only variable atoms, atoms with the same predicate symbol are always unifiable. We define the rules with which lemmas are derived. Variable merging and disjunction resolution are standard. The existential resolution rules are different from the standard paramodulation/superposition rules.
We assume that premises in geometric formulas can be freely permuted. As a consequence, in disjunction resolution, the resolved atoms are not necessarily the first atoms in the lemmas. We always assume that lemmas are normalized in the following ways:

1. Quantified variables that do not occur in the body are removed.
2. Repeated quantifications over the same variable are removed.
3. Identical copies of atoms are removed. This includes the case where the formula contains two orientations $v \not\approx w$ and $w \not\approx v$ of a disequality.

Definition 12. *We the rules of the lemma calculus. When a rule has more than one premise, we implictly assume that variables in the premises are renamed, s.t. no two premises have a variable in common.*

variable merging: *Let* $\lambda = \forall \overline{x}\ \ A_1(\overline{x}) \wedge \cdots \wedge A_p(\overline{x}) \wedge x_1 \not\approx x_1' \wedge \cdots \wedge x_q \not\approx x_q' \to \bot$. *Let* Σ *be a substitution of form* $x := x'$, *for two distinct* $x, x' \in \overline{x}$. *Then the following lemma is a* variable merging *of* λ :

$$\forall\ \overline{x}\Sigma\ \ A_1(\overline{x})\Sigma \wedge \cdots \wedge A_p(\overline{x})\Sigma \wedge x_1\Sigma \not\approx x_1'\Sigma \wedge \cdots \wedge x_q\Sigma \not\approx x_q'\Sigma \to \bot.$$

disjunction resolution: *Let* $\rho = \ \forall \overline{x}\ \Phi(\overline{x}) \to B_1(\overline{x}) \vee B_2(\overline{x}) \vee \cdots \vee B_q(\overline{x})$ *be a disjunctive formula. Let*

$$\lambda = \forall \overline{y}\ D_1(\overline{y}) \wedge D_2(\overline{y}) \wedge \cdots \wedge D_r(\overline{y}) \wedge y_1 \not\approx y_1' \wedge \cdots \wedge y_s \not\approx y_s' \to \bot$$

be a lemma. Assume that $B_1(\overline{x})$ *and* $D_1(\overline{y})$ *are unifiable with mgu* Σ. *Then*

$$\forall\ \overline{x}\Sigma\ \ \overline{y}\Sigma\ \ \Phi(\overline{x})\Sigma \wedge D_2(\overline{y})\Sigma \wedge \cdots \wedge D_r(\overline{y})\Sigma \wedge$$

$$y_1\Sigma \not\approx y_1'\Sigma \wedge \cdots \wedge y_s\Sigma \not\approx y_s'\Sigma \to B_2(\overline{x})\Sigma \vee \cdots \vee B_q(\overline{x})\Sigma$$

is a disjunction resolvent of ρ *with* λ.

existential resolution: *Let* $\rho = \forall \overline{x}\ \Phi(\overline{x}) \to \exists y\ B(\overline{x}, y)$ *be an existential formula. Let*

$$\lambda = \forall \overline{z}\ \forall v\ \Psi(\overline{z}) \wedge B(\overline{z}, v) \wedge v \not\approx z_1 \wedge \cdots \wedge v \not\approx z_s \to \bot$$

be a lemma, such that $z_1, \ldots, z_s \in \overline{z}$, *and* $v \notin \overline{z}$. *Assume that* $B(\overline{x}, y)$ *and* $B(\overline{z}, v)$ *have most general unifier* Σ. *Furthermore, assume that* $y\Sigma = v\Sigma,\ \ y\Sigma \notin \overline{x}\Sigma,\ \ v\Sigma \notin \overline{z}\Sigma$. *Then*

$$\forall\ \overline{x}\Sigma\ \ \overline{z}\Sigma\ \ \Phi(\overline{x})\Sigma \wedge \Psi(\overline{z})\Sigma \to B(\overline{z}, z_1)\Sigma \vee \cdots \vee B(\overline{z}, z_s)\Sigma$$

is an existential resolvent *of* ρ *with* λ. *Note that in case* $s = 0$, *the existential resolvent is a lemma. Otherwise it is a disjunctive rule.*

existential resolution (degenerated): *Let* $\rho = \forall \overline{x}\ \Phi(\overline{x}) \to \exists y\ B(\overline{x}, y)$ *be an existential formula. Let*

$$\lambda = \forall \overline{z}\ \forall v\ \Psi(\overline{z}) \wedge v \not\approx z_1 \wedge \cdots \wedge v \not\approx z_s \to \bot$$

be a lemma, such that $z_1, \ldots, z_s \in \overline{z}$, *and* $v \notin \overline{z}$. *Then*

$$\forall\ \overline{x}\ \overline{z}\ \Phi(\overline{x}) \wedge \Psi(\overline{z}) \to B(\overline{x}, z_1) \vee \cdots \vee B(\overline{x}, z_s)$$

is a (degenerated) existential resolvent of ρ *with* λ. *In case* $s = 0$, *the degenerated existential resolvent is a lemma. Otherwise, it is a disjunctive rule.*

Disjunction resolution will be used only as hyperresolution rule. For a given rule ρ, the algorithm will find q lemmas, and resolve all the conclusions away at once. Note that it is not possible to resolve upon a disequality. Variable merging will be used only in combination with the other rules. In existential resolution, the conjunction $\Phi(\overline{z})$ may contain disequalities. However, these disequalities are not allowed to contain v, because v cannot occur in $\Phi(\overline{z})$ at all. We give examples of disjunction resolution:

Example 3. $\forall x\ A(x) \to \exists y\ P(x,y),\ \ \forall z \forall t\ B(z) \wedge P(z,t) \to \bot \Rightarrow$
$\forall z\ A(x) \wedge B(x) \to \bot.$
$\forall x\ A(x) \to \exists y\ P(x,y),\ \ \forall z \forall t\ B(z) \wedge P(z,t) \wedge z \not\approx t \to \bot \Rightarrow$
$\forall z\ A(x) \wedge B(x) \to P(x,x).$
$\forall x_1 x_2\ \exists y\ F(x_1, y, x_2, y),$
$\forall z_1 z_2 \forall t\ G(z_1, z_2) \wedge F(z_1, t, z_2, t) \wedge z_1 \not\approx t \wedge z_2 \not\approx t \to \bot \Rightarrow$
$\forall x_1 x_2\ G(x_1, x_2) \to F(x_1, x_1, x_2, x_1) \vee F(x_1, x_2, x_2, x_2).$

Theorem 4. *The rules of Definition 12 are sound.*

Proof. Lemma factoring and disjunction resolution are standard rules. The correctness of existential resolution can be seen as follows: The lemma λ is equivalent to $\lambda' = \forall \overline{z}\ \Psi(\overline{z}) \to \forall v[\ B(\overline{z}, v) \to v \approx z_1 \vee \cdots \vee v \approx z_s]$. This implies $\lambda'' = \forall \overline{z}\ \Psi(\overline{z}) \to [\exists v\ B(\overline{z}, v)] \to B(\overline{z}, z_1) \vee \cdots \vee B(\overline{z}, z_s)$. Now $\exists v\ B(\overline{z}, v)$ can resolve with $\exists y\ B(\overline{x}, y)$ in ρ.
In the case of degenerated existential resolution, λ is equivalent to $\lambda' = \forall \overline{z}\, \Psi(\overline{z}) \to \forall v[v \approx z_1 \vee \cdots \vee v \approx z_s]$. From this, the conclusion can be derived by case analysis.

We call the improved algorithm S_m. It uses the rules of Definition 12 in order to compute closing lemmas. Whenever an interpretation (E, M) cannot be extended to an interpretation that makes G true, $S_m(G, E, M)$ returns a closing lemma of (E, M).
We explain how the algorithm of Definition 10 is modified: In case 1, $S_m(G, E, M)$ can simply return the lemma $\forall \overline{x}\ \phi_1(\overline{x}) \wedge \cdots \wedge \phi_p(\overline{x}) \to \bot$. In case 4, nothing needs to be changed. For cases 2 and 3, we will show that when the recursive calls have returned closing lemmas, $S_m(G, E, M)$ is able to construct a closing lemma for (E, M). In case 2, it uses disjunction resolution. In case 3, it uses one step of (possibly degenerated) existential resolution and several steps of disjunction resolution. Before we can prove this, we need to establish two properties of variable merging.

Lemma 2. *Let $\forall \overline{x}\ \Phi(\overline{x}) \to \bot$ be a lemma. Let Θ be a ground substitution, for which there exist distinct variables $x_1, \ldots, x_m \in \overline{x}$, such that $x_1\Theta = \cdots = x_m\Theta$. Then one can obtain in $m-1$ variable merging steps, a lemma of form $\forall\ (\overline{x}\Sigma)\ \Phi(\overline{x})\Sigma \to \bot$, for which there exists a ground substitution Θ', such that $\Theta = \Sigma \cdot \Theta'$ and the new lemma contains exactly one variable x' for which $x'\Theta' = x_1\Theta = \cdots x_m\Theta$.*

Proof. It follows from the definition of most general unifier, because Σ is the mgu of x and x'.

Lemma 3. *Let $\forall \overline{x}\ \Phi(\overline{x}) \to \bot$ be a lemma. Let Θ be a ground substitution, for which there exist some non-disequality atoms $A_1(\overline{x}), \ldots, A_m(\overline{x}) \in \Phi(\overline{x})$, s.t. $A_1(\overline{x})\Theta = \cdots = A_m(\overline{x})\Theta$. Then one can obtain, by iterated variable mergings, a lemma $\forall\ (\overline{x}\Sigma)\ \Phi(\overline{x})\Sigma \to \bot$, for which there exists a ground substitution Θ', s.t. $\Theta = \Sigma \cdot \Theta'$, and $\Phi(\overline{x})\Sigma$ contains exactly one atom B, s.t.*
$B\Theta' = A_1(\overline{x})\Theta = \cdots = A_m(\overline{x})\Theta$.

Proof. The proof is by induction on the number of distinct variables in $A_1(\overline{x}), \ldots, A_m(\overline{x})$. If $m = 1$, then there is nothing to prove. Otherwise, we have $m > 1$, and there exists a position on which $A_1(\overline{x})$ and $A_2(\overline{x})$ contain a different variable. Call the variables x_1 and x_2 (So one can write $A_1(\overline{x}) = A_1(\ldots, x_1, \ldots)$) and $A_2(\overline{x}) = A_2(\ldots, x_2, \ldots)$. Apply Lemma 2 with x_1 and x_2. In the resulting variable merging, the number of variables has decreased by one.

The following theorem guarantees that S_t can be modified to return a closing lemma in case 2.

Theorem 5. *Suppose that we have:*

1. *An interpretation* (E, M).
2. *A disjunctive formula* $r = \forall \overline{x}\ \Phi(\overline{x}) \rightarrow B_1(\overline{x}) \vee \cdots \vee B_q(\overline{x})$, *which is applicable to* (E, M) *with ground substitution* Θ.
3. *For each* $j,\ 1 \leq j \leq q,\quad$ *a closing lemma* λ_j *of* $(\ E,\ M + B_j(\overline{x})\Theta\ )$.

Then one can construct a closing lemma for (E, M) *using at most* q *applications of disjunction resolution. (and possibly several variable mergings)*

Proof. If we are lucky, one of the closing lemmas λ_j also closes (E, M). Otherwise, we show by induction on q that a closing lemma of (E, M) can be constructed in at most q disjunction resolution steps.
First consider λ_1. Write λ_1 in the form $\forall \overline{y}\ \Psi(\overline{y}) \rightarrow \bot$. Let Θ_1 be the ground substitution with which λ_1 closes $(\ E, M + B_1(\overline{x})\Theta\ )$. Because λ_1 does not close (E, M), there must be some atoms $A_1(\overline{y}), \ldots, A_m(\overline{y})$ in $\Psi(\overline{y})$, such that $A_j(\overline{y})\Theta_1 = B_1(\overline{x})\Theta$. According to Lemma 3, one can obtain by repeated variable merging a lemma λ_1' of form $\forall \overline{y}'\ A(\overline{y}') \wedge \Psi'(\overline{y}) \rightarrow \bot$, for which there exists a ground substitution Θ_1', s.t.

$$A(\overline{y}')\Theta_1' = B_1(\overline{x})\Theta \text{ and } \Psi'(\overline{y}')\Theta_1' \subseteq M.$$

Because $A(\overline{y}')$ and $B_1(\overline{x})$ are unifiable with mgu Σ, one can construct the following disjunction resolvent from r and λ_1' :

$$\rho = \forall\ \overline{x}\Sigma\ \ \overline{y}'\Sigma\ \ \Phi(\overline{x})\Sigma \wedge \Psi'(\overline{y}')\Sigma \rightarrow B_2(\overline{x})\Sigma \vee \cdots \vee B_q(\overline{x})\Sigma.$$

Because Σ is the mgu of $A(\overline{y}')$ and $B_1(\overline{x})$, there is a ground substitution Θ_1', s.t. $\Theta_1 = \Sigma \cdot \Theta_1'$. As a consequence ρ is applicable to (E, M). If $q = 1$, then ρ is a closing lemma. Otherwise, it can be checked that ρ and $\lambda_2, \ldots, \lambda_q$ satisfy the conditions $(1) \cdots (3)$ with $q - 1$, so that it is possible to apply induction.

The following guarantees that S_t can be modified to return a closing lemma in case 3.

Theorem 6. *Suppose that we have:*

1. *An interpretation* (E, M).
2. *An existential formula* $r = \forall \overline{x}\ \Phi(\overline{x}) \rightarrow \exists y\ B(\overline{x}, y)$, *which is applicable to* (E, M) *with ground substitution* Θ.

3. *For each* $e \in E$, *a closing lemma* λ_e *of* $(E, M + B(\overline{x}, y)\Theta_e)$, *with* $\Theta_e = \Theta + (y := e)$.
4. *For some* $\hat{e} \notin E$, *a closing lemma* $\lambda_{\hat{e}}$ *of* $(E + \hat{e},\ M + B(\overline{x}, y)\Theta_{\hat{e}})$, *with* $\Theta_{\hat{e}} = \Theta + (y := \hat{e})$.

Then it is possible to construct a closing lemma for (E, M) *using at most one application of existential resolution (possibly degenerated) and at most* $|E|$ *applications of disjunction resolution.*

Proof. Again, if we are lucky, one of the closing lemmas λ_e with $e \in E$, or $\lambda_{\hat{e}}$ with $\hat{e} \notin E$ already closes (E, M).
Otherwise consider $\lambda_{\hat{e}}$. Let Θ_λ be the substitution with which $\lambda_{\hat{e}}$ closes $(E + \hat{e}, B(\overline{x}, y)\Theta_{\hat{e}})$. Write $\lambda_{\hat{e}}$ in the form $\lambda_{\hat{e}} = \forall \overline{z}\ \overline{v}\ \Psi(\overline{z}, \overline{v}) \rightarrow \bot$, where $\overline{v}$ are the variables v for which $v\Theta_\lambda = \hat{e}$ and $\overline{z}$ are the remaining variables.
Because $\lambda_{\hat{e}}$ closes $(E + \hat{e}, M + B(\overline{x}, y)\Theta_{\hat{e}})$ but not (E, M), the sequence of variables $\overline{v}$ is not empty.
In case $\overline{v}$ contains more than one variable, one can apply Lemma 2 and obtain a new closing lemma $\lambda_{\hat{e}} = \forall \overline{z} v\ \Psi(\overline{z}, v) \rightarrow \bot$, which closes $(E + \hat{e}, M + B(\overline{x}, y)\Theta_{\hat{e}})$ with a ground substitution Θ_λ for which v is the only variable with $v\Theta'_\lambda = \hat{e}$.
The sequence of atoms $\Psi(\overline{z}, v)$ contains at least one atom containing v. We show that atoms containing v are either disequalities, or atoms of form $B(\overline{z}, v)$ with $B(\overline{z}, v)\Theta_\lambda = B(\overline{x}, y)\Theta_{\hat{e}}$. Let $A(\overline{z}, v)$ be a non-disequality atom that contains v. Then the instance $A(\overline{z}, v)\Theta_\lambda$ contains $\hat{e}$. Since M does not contain $\hat{e}$, and $A(\overline{z}, v)\Theta_\lambda \in M + B(\overline{x}, y)\Theta_{\hat{e}}$, it must be the case that $B(\overline{z}, v)\Theta_\lambda = B(\overline{x}, y)\Theta_{\hat{e}}$.
In case there is more than one atom $A(\overline{z}, v)$ containing v, for all of them holds that $A(\overline{z}, v)\Theta_\lambda = B(\overline{x}, y)\Theta_{\hat{e}}$. Therefore, we can apply Lemma 3 and obtain a new lemma $\lambda_{\hat{e}}$ which contains only one non-disequality atom which contains v. We distinguish two cases, dependent on whether $\Psi(\overline{z}, v)$ contains a non-disequality atom $B(\overline{z}, v)$ which is matched into $B(\overline{x}, y)\Theta_{\hat{e}}$.

1. $\lambda_{\hat{e}}$ can be written in the form $\forall \overline{z}\ v\ \Psi(\overline{z}) \wedge B(\overline{z}, v) \wedge v \not\approx z_1 \wedge \cdots \wedge v \not\approx z_s \rightarrow \bot$. We have assumed that the existential rule r and the lemma $\lambda_{\hat{e}}$ have no variables in common. Hence we can define $\Theta_{\lambda,r} = \Theta_\lambda + \Theta_{\hat{e}}$. We have

$$\Phi(\overline{x})\Theta_{\lambda,r} \subseteq M, \quad \Psi(\overline{z})\Theta_{\lambda,r} \subseteq M,$$

$$B(\overline{x}, y)\Theta_{\lambda,r} = B(\overline{x}, y)\Theta_{\hat{e}}, \quad B(\overline{z}, v)\Theta_{\lambda,r} = B(\overline{x}, y)\Theta_{\hat{e}}.$$

   Let Σ be the mgu of $B(\overline{x}, y)$ and $B(\overline{z}, v)$. There exists a ground substitution Θ_{rest}, s.t. $\Theta_{\lambda,r} = \Sigma \cdot \Theta_{rest}$.
   We know that $B(\overline{x}, y)$ contains y, and that $y\Theta_{\lambda,r} = \hat{e}$. Because v is the only variable in $B(\overline{z}, v)$ with $v\Theta_{\lambda,r} = \hat{e}$, it must the case that $B(\overline{z}, v)$ contains v at every position where $B(\overline{x}, y)$ contains y. From this it follows that $v\Sigma = y\Sigma$.
   In order to show that $y\Sigma \notin \overline{x}\Sigma$ and $v\Sigma \notin \overline{z}\Sigma$, it is sufficient to observe that $y\Theta_{\lambda,r} \notin \overline{x}\Theta_{\lambda,r}$ and $v\Theta_{\lambda,r} \notin \overline{z}\Theta_{\lambda,r}$.
   We can apply existential resolution, and obtain the rule

$$\forall\ \overline{x}\Sigma\ \overline{z}\Sigma\ \Phi(\overline{x})\Sigma \wedge \Psi(\overline{z})\Sigma \rightarrow B(\overline{z}, z_1)\Sigma \vee \cdots \vee B(\overline{z}, z_s)\Sigma.$$

Because of the fact that $\Theta_{\lambda,r} = \Sigma \cdot \Theta_{rest}$, we have

$$(\Phi(\overline{x})\Sigma)\Theta_{rest} \subseteq M, \quad (\Psi(\overline{z})\Sigma)\Theta_{rest} \subseteq M.$$

In case that $s = 0$, we already have a closing lemma. Otherwise, we will show that it is possible to apply Theorem 5.
First we show that for each j, $1 \leq j \leq s$, there is an $e \in E$, such that $B(\overline{z}, z_j)\Theta_\lambda = B(\overline{x}, y)\Theta_e$.
The left hand side $B(\overline{z}, z_j)\Theta_\lambda$ equals $B(\overline{z}\Theta_\lambda, z_j\Theta_\lambda)$. This in turn is equal to $B(\overline{x}\Theta, z_j\Theta_\lambda)$. If we define $e = z_j\Theta_\lambda$, (which is in E), then $B(\overline{x}\Theta, z_j\Theta_\lambda)$ is equal to $B(\overline{x}\Theta, y[y := e])$. Because $y \notin \overline{x}$, this equals $B(\overline{x}, y)(\Theta + (y := e))$, which in tuern equals $B(\overline{x}, y)\Theta_e$.
As a consequence each $B(\overline{z}, z_j)\Theta_{\lambda,r}$ is not true in (E, M). Hence the existential resolvent is applicable in (E, M). Because for each z_j, there is an $e \in E$, such that $B(\overline{z}, z_j)\Theta_{\lambda,r} = B(\overline{x}, y)\Theta_e$, and we have a closing lemma λ_e of $(E, M + B(\overline{z}, z_j)\Theta_{\lambda,r})$. This ensures that we can apply Theorem 5 and obtain a closing lemma for (E, M).

2. $\lambda_{\hat{e}}$ can be written in the form $\forall \overline{z}\ v\ \Psi(\overline{z}) \wedge v \not\approx z_1 \wedge \cdots \wedge v \not\approx z_s \rightarrow \bot$. We have assumed that the existential rule r and the lemma $\lambda_{\hat{e}}$ have no variables in common. We can construct the degenerated existential resolvent

$$\forall\ \overline{x}\ \overline{z}\ \Phi(\overline{x}) \wedge \Psi(\overline{z}) \rightarrow B(\overline{x}, z_1) \vee \cdots \vee B(\overline{x}, z_s).$$

It is easily checked that this rule is applicable on (E, M) with ground substitution $\Theta_\lambda + \Theta_{\hat{e}}$.
By similar reasoning as in the the previous case, we can see that Theorem 5 can be applied.

Note that the fact that $S_m(G, E, M)$ can always return a closing lemma, as a side effect implies that the calculus of Definition 12 is complete when used as a saturation calculus. The algorithm $S_m(G, E, M)$ can be made complete for finite-models by applying depth-first search. Because the learnt lemmas are kept, the loss in efficiency can be expected to be much less than the loss of efficiency that $S_t(G, E, M)$ would have. We give an example that shows that $S_m(G)$ improves over $S_t(G)$.

Example 4. Consider the set of rules $G = \ \exists x\ P_0(x), \quad \exists x\ P_1(x), \quad \exists x\ P_2(x), \exists x\ P_3(x), \quad \forall x_0 x_1 x_2 x_3\ P_0(x_0) \wedge P_1(x_1) \wedge P_2(x_2) \wedge P_3(x_3) \rightarrow \bot$.
$S_t(G)$ will consider a large sequence of interpretations of form:

$$\begin{array}{l}
(e_0,\ P_0(e_0) + P_1(e_0) + P_2(e_0) + P_3(e_0)), \\
(e_0 + e_1,\ P_0(e_0) + P_1(e_0) + P_2(e_0) + P_3(e_1)), \\
(e_0 + e_1,\ P_0(e_0) + P_1(e_0) + P_2(e_1) + P_3(e_0)), \\
(e_0 + e_1,\ P_0(e_0) + P_1(e_0) + P_2(e_1) + P_3(e_1)), \\
\ldots \\
(e_0 + e_1 + e_2,\ P_0(e_0) + P_1(e_1) + P_2(e_2) + P_3(e_0)), \\
(e_0 + e_1 + e_2,\ P_0(e_0) + P_1(e_1) + P_2(e_2) + P_3(e_1)), \\
(e_0 + e_1 + e_3,\ P_0(e_0) + P_1(e_1) + P_2(e_2) + P_3(e_2)), \\
\ldots \\
(e_0 + e_1 + e_2 + e_3,\ P_0(e_0) + P_1(e_1) + P_2(e_2) + P_2(e_3)).
\end{array}$$

The algorithm $S_m(G)$ will close the first interpretation $(e_0, P_0(e_0) + P_1(e_0) + P_2(e_0) + P_3(e_0))$ with rule $\forall x_0 x_1 x_2 x_3\ P_0(x_0) \wedge P_1(x_1) \wedge P_2(x_2) \wedge P_3(x_3) \rightarrow \bot$. Using existential resolution with $\exists x\ P_3(x)$ it will derive the lemma $\eta_1 = \forall x_0 x_1 x_2\ P_0(x_0) \wedge P_1(x_1) \wedge P_2(x_2) \rightarrow \bot$. Since η_1 closes $(e_0, P_0(e_0) + P_1(e_1) + P_2(e_2))$, it will apply one more time existential resolution, and derive $\eta_2 = \forall x_0 x_1\ P_0(x_0) \wedge P_1(x_1) \rightarrow \bot$. Continuing, it will derive $\eta_4 = \top \rightarrow \bot$ which closes $(\emptyset, \emptyset)$.

How many elements actually need to be tried, in addition to the first one, depends on the number s of disequalities in the first closing lemma $\forall \overline{x} y\ \Psi(\overline{x}) \wedge B(\overline{x}, y) \rightarrow y \not\approx x_1 \wedge \cdots \wedge y \not\approx x_s \rightarrow \bot$. Only those elements for which one of the disequalities is false, need to be tried. In the example, we always had $s = 0$.

6 Conclusions and Future Work

We have introduced a calculus, which is refutationally complete for first-order logic with equality. It can be implemented in such a way (using breadth-first search) that it is complete for finding finite models. In addition, it can be expected that our calculus will be good at handling problems containing *partial functions*.

We are in the process of implementing the calculus and studying refinements. The most successful refinement is *functional reduction*. It is best explained from an example. If one has a rule r of form $r = \forall \overline{x}\ \Phi(\overline{x}) \rightarrow \exists y\ P(\overline{x}, y)$, and there are no other positive occurrences of P in the set of formulas G, then P will be always functional in all attempted interpretations, because $\exists y\ P(\overline{x}, y)$ is extended only when it is false. As a consequence, whenever some lemma contains among its premises two occurrences of P of form $P(\overline{x}, y_1)$ and $P(\overline{x}, y_2)$, one can unify y_1 and y_2 without losing possible applications. This has turned out a good refinement, which improves performance by a factor of one hundred, especially on harder problems.

Another refinement that we tried is *lemma subsumption*. In analogy to resolution, one can define that λ_1 *subsumes* λ_2 if there exists a variable substitution Σ, s.t. $\lambda_1 \Sigma \subseteq \lambda_2$. Due to the way $S_m(G)$ operates, only backward subsumption needs to be considered. Our experiments show that approximately one fourth of the lemmas can be deleted with backward subsumption.

In future work, we intend to study other refinements, prove that they do not increase proof length whenever this is possible, and extend our experiments. We intend to take part in CASC with our implementation.

In addition, we would also like to obtain theoretical results that compare our calculus to superposition.

7 Acknowledgements

We thank Peter Baumgartner for reading a draft version of this report, and for discussions in the early stages of this work. Thanks also to Marc Bezem because

the idea for this research came after attending a talk about [3]. We thank him also for useful discussions on the topic of geometric logic.

References

1. Leo Bachmair and Harald Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994.
2. Peter Baumgartner and Cesare Tinelli. The model evolution calculus. In Franz Baader, editor, *CADE-19 - 19th International Conference on Automated Deduction*, volume 2741 of *LNAI*, pages 350–364. Springer, 2003.
3. Marc Bezem. Disproving distributivity in lattices using geometric logic. In *Workshop on Disproving, Non-Theorems, Non-Validity, Non-Provability*, pages 24–31. informal proceedings, July 2005.
4. Marc Bezem and Thierry Coquand. Automating coherent logic. In Geoff Sutcliffe and Andrei Voronkov, editors, *LPAR*, volume 3835 of *LNCS*, pages 246–260. Springer, 2005.
5. François Bry and Sunna Torge. A deduction method complete for refutation and finite satisfiability. In Jürgen Dix, Luis Fariñas del Cerro, and Ulrich Furbach, editors, *JELIA*, volume 1489 of *LNCS*, pages 122–138. Springer Verlag, 1998.
6. Hans de Nivelle and Maarten de Rijke. Deciding the guarded fragments by resolution. *Journal of Symbolic Computation*, 35(1):21–58, January 2003.
7. H. Ganzinger and H. de Nivelle. A superposition decision procedure for the guarded fragment with equality. In *LICS'99*, pages 295–303, 1999.
8. Yevgeny Kazakov and Hans de Nivelle. A resolution decision procedure for the guarded fragment with transitive guards. In Michaël Rusinowitch and David Basin, editors, *Second International Joint Conference on Automated Reasoning (IJCAR 2004)*, volume 3097 of *Lecture Notes in Artificial Intelligence*, pages 122–136, Cork, Ireland, July 2004. Springer Verlag.
9. Sun Kim and Hantao Zhang. ModGen: Theorem proving by model generation. In Barbara Hayes-Roth and Richard Korf, editors, *Proceedings of AAAI-94*, pages 162–167, 1994.
10. William McCune. Models and counter examples MACE2 (system). `http://www-unix.mcs.anl.gov/AR/mace2/` .
11. J. A. Robinson. A machine oriented logic based on the resolution principle. *Journal of the ACM*, 12:23–41, 1965.
12. John Slaney. Scott: A model-guided theorem prover. In *IJCAI-93*, pages 109–114, 1993.
13. Jian Zhang. Constructing finite algebras with FALCON. *Journal of Automated Reasoning*, 17:1–22, 1996.
14. Jian Zhang and Hantao Zhang. SEM: a system for enumerating models. In *IJCAI*, pages 298–303, 1995.
15. Lintao Zhang and Sharad Malik. The quest for efficient boolean satisfiability solvers. In Andrei Voronkov, editor, *CADE-18*, volume 2392 of *LNAI*, pages 295–313. Springer, 1992.