A Model Elimination Calculus with Built-in Theories

Peter Baumgartner
Universität Koblenz
Institut für Informatik
Rheinau 3-4
5400 Koblenz

E-mail: peter@infko.uni-koblenz.de

The *model elimination calculus* is a linear, refutationally complete calculus for first order clause logic. We show how to extend this calculus with a framework for *theory reasoning*. Theory reasoning means to separate the knowledge of a given domain or theory and treat it by special purpose inference rules. We present two versions of theory model elimination: the one is called *total theory model elimination* (which allows e.g. to treat equality in a rigid E-resolution style), and the other is called *partial theory model elimination* (which allows e.g. to treat equality in a paramodulation style).

1. Introduction

The *model elimination calculus* (ME calculus) has been developed already in the early days of automated theorem proving (Lov78b). It is a linear, refutationally complete calculus for first order clause logic. In this paper, we will show how to extend model elimination with theory reasoning.

Technically, *theory reasoning* means to relieve a calculus from explicit reasoning in some domain (e.g. equality, partial orders) by taking apart the domain knowledge and treating it by special inference rules. In an implementation, this results in a universal "foreground" reasoner that calls a specialized "background" reasoner for theory reasoning. Theory reasoning comes in two variants (Sti85): *total* and *partial* theory reasoning. Total theory reasoning generalizes the idea of finding complementary literals in inferences (e.g. resolution) to a semantic level. For example, in theory resolution the foreground reasoner may select from some clauses the literal set $\{a < b, b < c, c < a\}$, pass it to the background reasoner (assume that $<$ is interpreted as a strict ordering, i.e. as a transitive and irreflexive relation) which in turn should discover that this set is contradictory. Finally the theory resolvent is built as in ordinary resolution by collecting the rest literals. The problem with total theory reasoning is that in general it cannot be predicted what literals and how many variants of them constitute a contradictory set. As a solution, partial theory

reasoning tries to break the "big" total steps into more managable smaller steps. In the example, the background reasoner might be passed $\{a < b, b < c\}$, compute the logical consequence $a < c$ and return it as a new subgoal, called "residue", to the foreground reasoner. In the next step, the foreground reasoner might call the background reasoner with $\{a < c, c < a\}$ again, which detects a trivial contradiction and thus concludes this chain. It is this *partial* theory reasoning we are mostly interested in.

Theory reasoning is a very general scheme and thus has many applications, among them are *reasoning with taxonomical knowledge* as in the Krypton system (BGL85), *equality reasoning* as by paramodulation or E-resolution, building in *theory-unification*, and building in the axioms of the "reachability" relation in the translation of modal logic to ordinary first order logic.

The advantages of theory reasoning, when compared with the naive method of supplying the theories's axioms as clauses, are the following: for the first, the theory inference system may be specially tailored for the theory to be reasoned with; thus higher efficiency can be achieved by a clever reasoner that takes advantage of the theories' properties. For the second, a lot of computation that is not relevant for the overall proof plan is hidden in the background. Thus proofs become shorter and are more compact, leading to better readability.

Of course, theory reasoning is not new. It was introduced by M. Stickel within the general, non-linear resolution calculus (Sti85; Sti83). Since then the scheme was ported to many calculi. It was done for matrix methods in (MR87), for the connection method in (Bib87; Pet90), and for the connection graph calculus in (Ohl86; Ohl87). In (Bau92a) we showed that total theory reasoning is compatible to ordering restrictions.

However there are significant differences between these works and the present one: for the first, model elimination is a *linear* calculus, which roughly means that an initially chosen goal clause is stepwisely processed until the refutation is found. Being a very efficient restriction, we want to keep it in our theory calculus. However none of the theory extensions of the above calculi makes use of linear restrictions. As a consequence we also need a new completeness proof and cannot use e.g. Stickel's proof. This new proof is our main result.

Another difference is our emphasis on partial theory reasoning. The completeness of the overall calculus depends from the completeness of the background reasoner for partial theory reasoning. Except Stickel (Sti85), the above authors do not supply sufficient completeness preserving criteria for the background reasoner. Again, since Stickel's calculus is nonlinear, his criteria cannot be applied in our case. Below we will define a reasonable criteria that meets our demands. This criteria also captures a treatment of equality by "linear paramodulation". Since linear paramodulation is complete (FHS89) we obtain as a corrollary the completeness of model elimination with paramodulation.

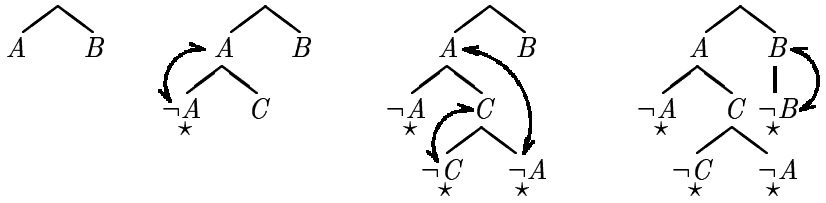
We will differ from Loveland's original ME calculus in two aspects: for the first, we have omitted some efficiency improvements such as factoring, and also we do not disallow inference steps that yield identical literals in a chain. This happens because in this paper we want to concentrate on the basic mechanisms of theory reasoning. We will adopt the efficiency improvements later. For the second we made a change in data structures: instead of *chains* we follow (LSBB92) and work in a tree-like setting in the tradition of analytic tableaux. This happens because we are mostly interested

to implement our results in the SETHEO theorem prover(see also (LSBB92)), which is based on that tableaux.

2. A Brief Introduction to Model Elimination

As mentioned above, we will follow the lines from (LSBB92) and define the inference rules as tree-transforming operators. Since we should not assume this format to be well-known, we will supply a brief and informal introduction.

In our format, model elimination can be seen as a restriction of semantic tableaux with unification for clauses (see (Fit90)). This restriction will be explained below. A *tableau* is, roughly, a tree whose nodes are labelled with literals in such a way that brother nodes correspond to a clause in the given clause set. A *refutation* is the construction of a tableau where every branch is contradictory. For this construction we have to start with an initial tableau consisting of a single clause, and then repeatedly apply the inference rules *extension* and *reduction* until every branch is checked to be contradictory. Consider the unsatisfiable clause set $\{A \vee B, \neg A \vee C, \neg C \vee \neg A, \neg B\}$. and the following refutation:



1. Initial tableau with $A \vee B$
2. Extension with $\neg A \vee C$
3. Extension with $\neg C \vee \neg A$
4. Reduction
5. Extension with $\neg B$

In step 1, the tableau consisting of $A \vee B$ is built. In step 2 the branch ending in A is extended with $\neg A \vee C$ and marked with a $\star$ (such branches are called *closed*); in general, extension is only allowed if the *leaf* of the branch is complementary to a literal in the clause extended with. Step 3 is an extension step with $\neg C \vee \neg A$, and the branch ending in $\neg C$ is closed. Step 4 depicts the reduction inference: a branch, in this case $A\,C\,\neg A$, may be closed if the leaf is complementary to one of its ancestors. Finally, in step 5 the last open branch is closed by extension with $\neg B$.

Note that a closed branch contains complementary literals A and $\neg A$ and thus is unsatisfiable. If all branches are closed then the input clause set is unsatisfiable.

As usual, the ground case is lifted to the general case by taking variants of clauses, and establishing complementarism by means of a most general unifier. It should be noted that this unifier has to be applied to the entire tableau.

There is a close correspondance to linear resolution (see e.g. (CL73)): the set of open leafs corresponds to the near parent clause, extension corresponds to input resolution, and reduction corresponds to ancestor resolution. This correspondance also explains why model elimination is called "linear". If the restriction "the *leaf* (and not just any other literal in the branch) must be one of the complementary literals" is dropped, the calculus is no longer linear.

Lovelands original chain-notation (Lov78a) with A- and B-literals can be seen as a linear notation for our tableaux. More precisely, the open branches can bijectively be

mapped to a chain, where the leafs are B-literals and the inner nodes are A-literals. If in the tableau model elimination always the "rightmost" branch is selected for extension or reduction, then there exist corresponding inference steps in chain model elimination. See (BF92) for a detailled comparison.

3. Theory Unifiers

A *clause* is a multiset of literals written as $L_1 \vee \ldots \vee L_n$. A *theory* $\mathcal{T}$ is a satisfiable set of clauses.[1] Concerning model theory it is sufficient to consider Herbrand-interpretations only, which assign a fixed meaning to all language elements short of atoms; thus we define a *(Herbrand-) interpretation* to be any total function from the set of ground atoms to $\{true, false\}$. A *(Herbrand-) $\mathcal{T}$-interpretation* is an interpretation satisfying the theory $\mathcal{T}$. An interpretation (resp. $\mathcal{T}$-interpretation) I *satisfies* (resp. $\mathcal{T}$-satisfies) a clause set M iff I simultaneously assigns *true* to all ground instances of the clauses in M. ($\mathcal{T}$)-(un-)satisfiability and ($\mathcal{T}$-)validity of clause sets are defined on top of this notion as usual.

As with non-theory calculi the refutations should be computed at a most general level; this is usually achieved by most general unifiers. In the presence of theories however, unifiers need not be unique, and they are replaced by a more general concept:

Definition 3.1 Let $S = \{L_1, \ldots, L_n\}$ be a literal set. S is called *$\mathcal{T}$-complementary* iff the $\forall$-quantified disjunction $\forall(\overline{L_1} \vee \ldots \vee \overline{L_n})$ is $\mathcal{T}$-valid. We say that a substitution σ is a *$\mathcal{T}$-unifier* for S iff $S\sigma$ is $\mathcal{T}$-complementary. A substitution σ is *more general* than a substitution θ iff $\sigma \leq theta$ iff there exists a substitution δ such that $\sigma\delta|_{dom(\sigma)}$. A $\mathcal{T}$-unifier for a set S is *most general* iff there does not exist another $\mathcal{T}$-unifier for S which is more general.

A "partial" variant is as follows: a pair (σ, R), where σ is a substitution and R is a literal, is a *$\mathcal{T}$-residue of S* iff $S\sigma \cup \left\{\overline{R}\right\}$ is minimal $\mathcal{T}$-complementary. A pair (σ, R) is *more general* than a pair (θ, Q) iff $(\sigma, R) \leq (theta, Q)$ iff there exists a substitution δ such that $\sigma\delta|_{dom(\sigma)}$ and $R\delta = Q$. A $\mathcal{T}$-residue for a set S is *most general* iff there does not exist another $\mathcal{T}$-residue for S which is more general. **(End Definition)**

There is a subtle difference between the $\mathcal{T}$-complementary of a literal set and the $\mathcal{T}$-unsatisfiability of S when S is read as a set of unit clauses. These notions are the same only for ground sets. Consider, for example, a language with at least two constant symbols a and b and the "empty" theory $\emptyset$. Then $S = \{P(x), \neg P(y)\}$ is, when read as a clause set, $\emptyset$-unsatisfiable, but S is not $\emptyset$-complementary, because the clause $P(x) \vee \neg P(y)$ is not $\emptyset$-valid (because the interpretation with $I(P(a)) = false$ and $I(P(b)) = true$ is no model). However, when applying the MGU $\sigma = \{x \leftarrow y\}$ to S the resulting set $S\sigma$ is $\emptyset$-complementary.

The importance of "complementary" arises from its application in inference rules, such as resolution, which have for soundness reasons be built on top of "complementarism", but not on "unsatisfiablitity". Since we deal with theory inference rules, we

[1] This restriction is motivated by the intended application of a Herbrand-Theorem, which only holds for universally quantified theories

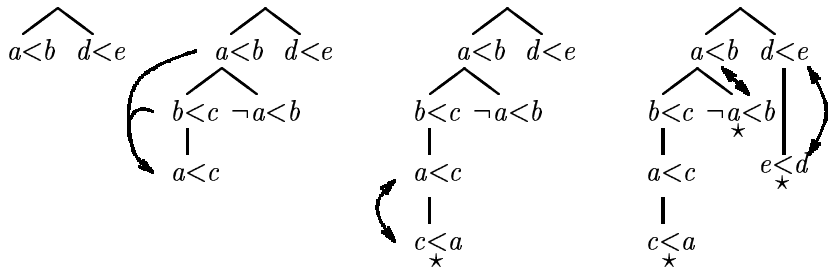
had to extend the usual notion of "complementarism" to "$\mathcal{T}$-complementarism". As an example consider the theory $\mathcal{E}$ of equality. Then $S = \{P(x), y = f(y), \neg P(f(f(a)))\}$ is $\mathcal{E}$-unsatisfiable but not $\mathcal{E}$-complementary. However with the $\mathcal{E}$-unifier $\sigma = \{x \leftarrow a, y \leftarrow a\}$, $S\sigma$ is $\mathcal{E}$-complementary. In this context it might be interesting to known that our notion of theory unifier generalizes the notion of *rigid E-unifier* (GNPS90) to more general theories than equality (see (Bau92a) for a proof).

The semantics of a residue (L, σ) of S is given as follows: L is a logical consequence of $S\sigma$; operationally L is a new goal to be proved. For example let $S' = \{P(x), y = f(y)\}$. Then $(\{x \leftarrow y\}, P(f(y)))$ is an $\mathcal{E}$-residue of S', since $S'\{x \leftarrow y\} \cup \{\neg P(f(y))\} =$
$\{P(y), y = f(y), \neg P(f(y))\}$ is minimal $\mathcal{E}$-complementary.

4. Calculus

Theory reasoning calculi require the computation of theory unifiers. Of course, any implementation of theory-unification in the traditional sense (see (Sie89)), e.g. AC-unification, performs a stepwise computation. Since we are interested in partial theory reasoning (see the introduction), this computation shall not remain hidden for the foreground reasoner; instead, intermediate results shall be passed back from the background reasoner to the foreground reasoner in the form of *residues*.

Let us informally describe this on the ground level with the aid of an example. Consider the clause set $S = \{a < b \vee d < e,\ b < c \vee \neg a < b,\ c < a,\ e < d\}$. S is unsatisfiable in the theory of strict orderings ($<$ is transitive and irreflexive), and this is a theory model elimination proof:



1. Initial tableau 2. Partial extension with $a<b \vee d<e$ with $b<c \vee \neg a<b$ 3. Total extension with $c<a$ 4. Reduction 5. Total extension with $e<d$

In step 1, the tableau consisting of $a < b \vee b < c$ is built. In step 2 the branch ending in $a < b$ is *partially extended* with the clause $b < c \vee \neg a < b$ and the *residue* $a < c$ (in this ground example no substitutions appear). The literals of the extending clauses which are relevant for the extension step, here solely $b < c$, are called *extending literals*. This step is sound, because $a < c$ is a logical consequence of its ancestors $a < b$ and $b < c$. The literals that semantically justify the inference step in this way are called the *key set* (here $\{a < b, b < c\}$). Since the branch resulting from this step is not contradictory, it is not closed (marked with a star). Besides partial extension, there exists another inference rule called *total extension*. Step 3. serves as an example: the extension with $c < a$ yields a theory-contradiction with $a < c$. Thus the branch may be closed. The ancestor literals that justify the total inference step, i.e. the contradictory set $\{c < a, a < c\}$ is also called a "key

set", and $c < a$ is also called "extending literal". Step 4. is an ordinary reduction step, and step 5 is a total extension step again.

The inference steps are restricted in such a way that their key sets must consist a) of the leaf and b) possibly some other literals of the old branch, and c) of all extending literals. Condition c) implies that *all* new clauses are needed, and condition b) is the generalization of the condition "the *leaf* must be one of the complementary literals" in non-theory model elimination (section 2) to theory model elimination.

Let us now come to a formal treatment. We are concerned with ordered, labelled trees with finite branching factor and finite length. A *branch b of length k* is a sequence $b = n_0 \circ n_1 \circ \cdots \circ n_k$ of nodes, where n_0 is the root, n_{i+1} is a son of n_i and n_k is a leaf, and a tree is represented as a multiset of branches. A *literal tree* is a tree whose nodes are labelled with literals, except the root, which remains unlabelled. For our purpose it is convenient to confuse a branch $n_0 \circ n_1 \circ \cdots \circ n_k$ with the sequence of its labels $L_1 \circ \cdots \circ L_k$ or with its literal set $\{L_1, \ldots, L_k\}$. A substitution is applied to a branch by applying it to its labels in the obvious way; similarly it is applied to a tree by application to all its branches. A literal tree T' is obtained from a literal tree T by *extension with a clause $L_1 \vee \ldots \vee L_n$ at a branch b* iff

$$T' = T - \{b\} \cup \{b \circ l_i \mid i = 1 \ldots n \text{ and } l_i \text{ is labelled with } L_i\}$$

In this case we also say that T' contains a clause $L_1 \vee \ldots \vee L_k$ rooted at b.

The term "to close a branch" means to attach an additional label "$\star$" to its leaf in order to indicate that the branch is proved to be $\mathcal{T}$ -complementary. A branch is *open* iff it is not labelled in that way.

Definition 4.1 ($\mathcal{T}$-model elimination) Let M be a clause set and $\mathcal{T}$ be a theory. An *initial model elimination tableau for M with top clause C* is a literal tree that results from extending the empty tree (the tree that contains only the empty branch) with the the clause C.

A *model elimination tableau (ME tableau) for M* is either an initial ME tableau or a literal tree obtained by a single application of one of the following inference rules to a ME tableau T:

Partial extension step: (cf. figure 1) Let $b = L_1 \circ \ldots \circ L_{k-1} \circ L_k$ be an open branch in T . Suppose there exist new variants $C_i = K_i^1 \vee \ldots \vee K_i^{m_i}$ $(i = 1 \ldots n)$ of clauses in M. These clauses are called the *extending clauses* and the sequence $K_1^1 \circ \cdots \circ K_n^1$ is called the *extending literals*.

In order to describe the appending of the extending clauses, we define the literal tree T_n and the "actual branch to extend", b_n, recursively as follows: if $n = 0$ then $T_0 := T$ and $b_0 := b$, else T_n is the literal tree obtained from T_{n-1} by extending with the clause C_n at b_{n-1} and $b_n := b_{n-1} \circ K_n^1$.

Let $\mathcal{K}$ be a subset of the literal set of b_n with $L_k, K_1^1, \ldots, K_n^1 \in \mathcal{K}$. Borrowing a notion from (Sti85), $\mathcal{K}$ is called the *key set*. If there exists a most general $\mathcal{T}$-residue (σ, R) of $\mathcal{L}$, then partial theory extension yields the tree T_n', where T_n' is obtained from $T_n\sigma$ by extension with the unit clause R at $b_n\sigma$.

Total extension step: This is similar to "partial extension step"; instead of appending a residue, the branch is closed. Let b, C_i, T_n and b_n and $\mathcal{K}$ as in

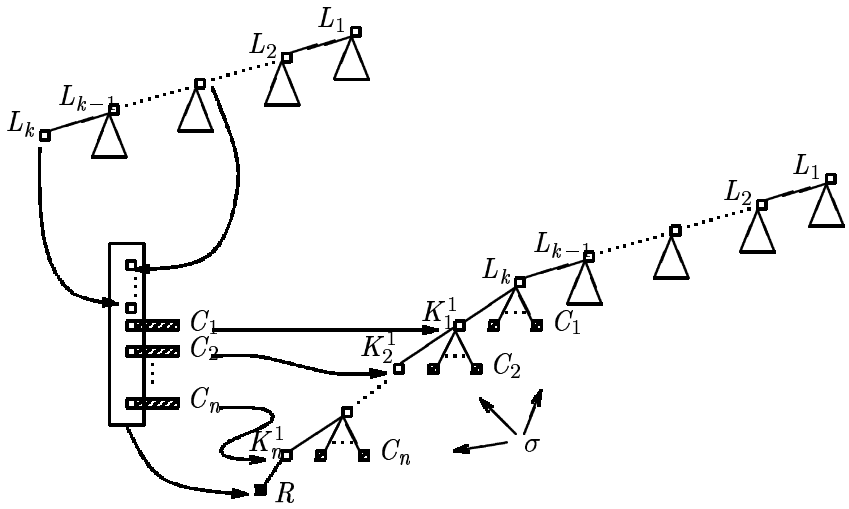


Figure 1: Partial extension step

"partial extension step". If there exists a most general $\mathcal{T}$-unifier σ for $\mathcal{K}$, and $\mathcal{K}\sigma$ is minimal $\mathcal{T}$-complementary, then total theory extension yields the literal tree $T_n\sigma$, and the branch $b_n\sigma \in T_n\sigma$ is closed.

A total extension step with $n = 0$ is also called **reduction step**.[2] A *derivation from M with top clause C and length n* is a finite sequence of ME tableaux $T_0, T_1, \ldots, T_n$, where T_0 is an initial tableau for M with top clause C, and for $i = 1 \ldots n$ T_i is the tableau obtained from T_{i-1} by one single application of one of the above inference rules with new variants of clauses from M. If additionally in T_n every branch is closed then this derivation is called a *refutation of M*. The *partial theory model elimination calculus (PTME-calculus)* consists of the inference rules "partial extension step" and "total extension step"; the *total theory model elimination calculus (TTME-calculus)* consists of the single inference rule "total extension step" **(End Definition)**

The key sets $\mathcal{K}$ play the role of a semantical justification of each step. The condition $L_k \in \mathcal{K}$ generalizes the condition "the *leaf* must be one of the complementary literals" from non-theory model elimination (section 2).

In practice it is important that the key sets and residues may be restricted to some typical, syntactical form. For example, if the theory is equality, and the calculus shall be instantiated with "paramodulation", then in the ground case the key sets $\mathcal{K}$ in partial steps are of the form $\mathcal{K} = \{L[t], t = u\}$[3] or $\mathcal{K} = \{L[t], u = t\}$, and the residues are of the form $(\emptyset, L[t \leftarrow u])$; in total steps it suffices to restrict the key set to the form $\mathcal{K} = \left\{L, \overline{L}\right\}$ or $\mathcal{K} = \{\neg a = a\}$. In lifting these paramodulation steps to the first-order level, it is neccessary to allow instantiating before paramodulation (see e.g. (FHS89)). For example, if $\{P(x, x), a = b\}$ is a key set, then for completeness reasons it might be neccessary to instantiate first with $x \leftarrow f(y)$, which

[2] This notion is kept for historical reasons

[3] $L[t]$ means that the term t occurs in the literal L, $L[t \leftarrow u]$ is the literal that results from replacing one occurence of t with u

yields $P(f(y),f(y))$ and then paramodulate into y which finally yields $P(f(b),f(a))$. Since instantiating is not neccessary in a non-linear setting, this example also shows that inference systems for partial theory reasoning must *in principle* be designed differently than for non-linear calculi.

It should be noted that the *order* of the extending clauses is immaterial for completeness.

5. Completeness

Besides soundness, which is usually easy to prove, *(refutational) completeness* is the most important demand for a logic calculus. In order to establish such a result for partial theory reasoning, the theory reasoning component must be taken care of.

In our viewpoint, the computation of the residues in partial extension steps, and the computation of the substitutions in total extension steps should be described by a "theory calculus" with two respective inference rules: the one derives from a key set a residue, and the other derives from a key set a theory-unifier. In order to establish the completeness of the overall calculus, the theory calculus itself must be "complete". However, in order to formulate this we do not want to fix to a certain calculus; instead we will use the following abstract characterization.

Definition 5.1 Let $\mathcal{T}$ be a theory, S_1 be a ground $\mathcal{T}$-unsatisfiable literal set and $L_0 \in S_1$. Then a *linear and liftable $\mathcal{T}$-refutation of S_1 with top literal L_0* consists of the three sequences

$$\begin{array}{llll} S_1, & \ldots\ S_n, & S_{n+1} & \text{and} \\ K_1, & \ldots\ K_n, & K_{n+1} & \text{and} \\ (L_1,\sigma_1), & \ldots\ (L_n,\sigma_n), & \sigma_{n+1} & \end{array}$$

such that for $i = 1 \ldots n$:

1. $L_{i-1} \in K_i$, $K_i \subseteq S_i$ and (L_i, σ_i) is a $\mathcal{T}$-residue of K_i.
2. $S_{i+1} := S_i\sigma_i \cup \{L_i\}$.
3. $L_n \in K_{n+1}$, $K_{n+1} \subseteq S_{n+1}$, and $K_{n+1}\sigma_{n+1}$ is minimal $\mathcal{T}$-complementary.
4. For every[4] $K'_i \leq K_i$ there exists a residue $(L'_i\sigma'_i)$ such that $L'_i \leq L_i$ and $\sigma'_i \leq \sigma_i$.
5. For every $K'_{n+1} \leq K_{n+1}$ there exists a $\mathcal{T}$ unifier $\sigma'_{n+1} \leq \sigma_{n+1}$ for K'_{n+1}.

Such a refutation is called *total* instead of linear iff $n = 0$. **(End Definition)**

The idea behind items 1. – 3. is to stepwisely modify an initially chosen goal L_0, leading to $L_1, L_2, \ldots, L_n$ until a contradiction is obvious (The presence of unifiable and syntactical complementary literals might be such a case, or, in equational reasoning, the presence of a literal $\neg s = t$ where s and t are unifiable). The K_is have the same meaning as the key sets in the definition of theory model elimination. In the first chain of the introductory example in the previous section, $n = 1$,

[4] $K' \leq K$ iff $\exists\delta : K'\delta = K$ and $\sigma' \leq \sigma$ iff $\exists\delta : \sigma'\delta|dom(\sigma) = \sigma$

$S_1 = \{a < b, b < c, c < a\}$, $L_0 = a < b$, $K_1 = \{a < b, b < c\}$, $(\sigma_1, L_1) = (\emptyset, a < c)$, $S_2 = \{a < b, b < c, c < a, a < c\}$, $K_2 = \{c < a, a < c\}$ and $\sigma_2 = \emptyset$. This strategy can also be roughly explained in linear resolution terminology: L_1 plays the role of the top clause, and the L_{i+1} are derived from the near parent L_i and a collection of far parent clauses $K_i \subseteq S_i$.

Note that although S_1 is a ground set, substitutions are involved. This is, because $\forall$-quantified variables might be introduced in the residues and thus the S_j $(j > 1)$ might no longer be ground. Items 4. is the lifting requirements for the residues, and item 5. is the lifting requirement for the concluding unifier.

Now we can turn to completeness of theory model elimination. In an attempted model elimination refutation it is essential to pick a suitable clause for the initial tableau. For example, if $S = \{A, B, \neg B\}$ then no proof can be found when the initial tableau is built from A. What we need is expressed in the completeness theorem:

Theorem 5.2 (Completeness of partial theory model elimination) *Let $\mathcal{T}$ be a theory. Suppose that for every minimal $\mathcal{T}$-unsatisfiable ground literal set S and every $L \in S$ there exists a linear and liftable $\mathcal{T}$-refutation of S with top literal L.*
Let M be a $\mathcal{T}$-unsatisfiable clause set. Let $C \in M$ be such that C is contained in some minimal $\mathcal{T}$-unsatisfiable subset of M. Then there exists a PTME refutation of M with top clause C.

The completeness of TTME follows as a corollary of PTME if the theory reasoner can find a $\mathcal{T}$-unifier in one step, or, more technically:

Corollary 5.3 *Let $\mathcal{T}$ be a theory. Suppose that for every minimal $\mathcal{T}$-unsatisfiable ground literal set S there exists a total and liftable $\mathcal{T}$-refutation of S.*
Let M be a $\mathcal{T}$-unsatisfiable clause set. Let $C \in M$ be such that C is contained in some minimal $\mathcal{T}$-unsatisfiable subset of M. Then there exists a TTME refutation of M with top clause C.

The completeness proof employs the standard technique of proving first the ground case and then lifting to the variable case. Thus we apply a theory-version of the Skolem-Herbrand-Gödel theorem. Such a theorem only holds for universally quantified formula. This fact explains our restriction to *clausal* theories.

For the ground proof we need the following notion: a clause $C \in M$ in an $\mathcal{T}$-unsatisfiable clause set M is called *essential in M* iff $M - \{C\}$ is not $\mathcal{T}$-unsatisfiable. Note that not every $\mathcal{T}$-unsatisfiable clause set has an essential literal, e.g. $\{A, \neg A, B, \neg B\}$ is $\emptyset$-unsatisfiable but deleting any element results in a still $\emptyset$-unsatisfiable set. However every literal in a *minimal* $\mathcal{T}$-unsatisfiable set is essential.

Lemma 5.4 *Let M be a $\mathcal{T}$-unsatisfiable ground clause set, with $L, L_1 \vee \ldots \vee L_n \in M$ being essential. Define $M_i = M - \{L_1 \vee \ldots \vee L_n\} \cup \{L_i\}$ (for $i = 1 \ldots n$). Then every M_i is $\mathcal{T}$-unsatisfiable and the clauses L and L_i are essential in some M_j $(1 \leq j \leq n)$.*

This lemma is needed in the proof of the following ground completeness lemma:

Lemma 5.5 *Let $\mathcal{T}$ be a theory and M be a $\mathcal{T}$-unsatisfiable ground clause set with essential clause C. Then there exists a PTME refutation of M with top clause C.*

Proof. For convenience some terminology is introduced: if we speak of "replacing a clause C_1 in a derivation by a clause C_2" we mean the derivation that results from replacing some specific occurence of C_1 by C_2, which must be a superset of $C1$, in every tableau in the derivation. By a "derivation of a clause $L_1 \vee \ldots \vee L_n$" we mean a derivation that ends in a tableau which in turn contains n open branches ending in brother leafs $L_1, \ldots, L_n$. Furthermore, if D_1 is a derivation of a clause C and D_2 is a derivation with top clause C, then by "appending D_1 and D_2" we mean the derivation that results from extending D_1 with the inferences of D_2 in order, where one specific occurence of C in D_1 takes the role of the top clause C in D_2.

Let $k(M)$ denote the number of occurences of literals in M minus the number of clauses in M ($k(M)$ is called the *excess literal parameter* in (AB70)). Now we prove the claim by induction on $k(M)$.

In the induction base $k(M) = 0$. Then M must be a set of unit clauses, i.e. a literal set. Now set $L_0 = C$ and consider definition (5.1). The sequences defined there can be mapped to a refutation as follows: the initial tableau consists of L_0. The $\mathcal{T}$-residues $(\sigma_1, L_1), \ldots, (\sigma_n, L_n)$ of the respective sets $K_1, \ldots, K_n$ are mapped to n partial extension steps as follows: in step i choose as the key set K_i, as extending literals $K_i - \{L_{i-1}\}$, and as residue (σ_i, L_i). A final total extension step with key set K_{n+1} and extending literals $K_{n+1} - \{L_n\}$ and substitution σ_{n+1} yields the desired refutation.

To complete the induction assume now that $k(M) > 0$ and that the result holds for sets M' with $k(M') < k(M)$. We need a further case analyses.

Case 1: C is a non-unit clause of the form $C = L \vee R_1 \vee \ldots \vee R_n$. Define

$$M'_L = (M - \{C\}) \cup \{L\} \quad (1)$$

$$M'_R = (M - \{C\}) \cup \{R_1 \vee \ldots \vee R_n\} \quad (2)$$

Both M'_L and M'_R are unsatisfiable, since otherwise a model for one of them were a model for M, which contradicts the assumption that M is unsatisfiable. Find a minimal $\mathcal{T}$-unsatisfiable $M_L \subset M'_L$ that contains L. Such a set must exist, because otherwise $M'_L - \{L\} \subset M$ were $\mathcal{T}$-unsatisfiable, and with $C \notin M'_L$ it follows that C is not essential in M. Since M_L is minimal $\mathcal{T}$-unsatisfiable L is essential in M_L. Since $k(M_L) < k(M)$ we can apply the induction hypothesis and obtain a refutation D_L of M_L with top clause L. We may assume that D_L is in the following normal form: in every extension or reduction step, L does not occur as an extending clause. Such a normal form can always be achieved, since L is the top clause, and thus in every step the extending clause L can be replaced by the ancestor clause L.

By the same argumentation as for L, the clause $R_1 \vee \ldots \vee R_n$ is essential in M_R, and since $k(M_R) < k(M)$ we can apply the induction hypothesis again and obtain a refutation D_R of M_R with top clause $R_1 \vee \ldots \vee R_n$.

Now replace in D_R every occurence of the clause $R_1 \vee \ldots \vee R_n$ by $L \vee R_1 \vee \ldots \vee R_n$. Call this derivation D'_R. D'_R is derivation of several occurences of a clause L from M with top clause $L \vee R_1 \vee \ldots \vee R_n$. Now append D'_R with D_L as many times until all these occurences of the clause L are closed. Since D_L is in the above normal form, the clause L is no longer used in this final derivation. Thus we obtain the desired refutation of M.

Case 2: C is a unit clause $C = K$. Since $k(M) > 0$, M contains a non-unit clause D. We distinguish two cases. In the first (and trivial) case, D is not essential

in M. Thus $M' = M - \{D\}$ is $\mathcal{T}$-unsatisfiable. But C is still essential in M', because otherwise it were not essential in M either. Since $k(M') < k(M)$ the refutation as claimed exists by the induction hypothesis.

In the other case D is essential in M. By lemma (5.4) D contains a literal L such that $M_L = (M - \{D\}) \cup \{L\}$ is $\mathcal{T}$-unsatisfiable, and K and L are essential in M_L. It holds that $k(M_L) < k(M)$. Thus by the induction hypothesis there exists a refutation D_K of M_L with top clause K. D is of the form $D = L \vee R_1 \vee \ldots \vee R_n$. Since L is essential in M_L and $k(M_L) < k(M)$ there exists by the induction hypothesis a refutation D_L of M_L with top clause L. As for D_L in case 1 above, the set D_L here can be assumed to be in the same normal form, i.e. L is not used as an extending clause in any inference step.

Let $M_R = (M - \{D\}) \cup \{R_1 \vee \ldots \vee R_n\}$. By the same argumentation as in case 1, M_R is $\mathcal{T}$-unsatisfiable and $R_1 \vee \ldots \vee R_n$ is essential in M_R, and by the induction hypothesis there exists a refutation D_R of M_R with top clause $R_1 \vee \ldots \vee R_n$.
Now we can put things together. First replace in D_K every occurence of the clause L by D. The result is a derivation D'_K of several occurences of a clause $R_1 \vee \ldots \vee R_n$ from M with top clause K. Now append D'_K with D_R as many times until all occurences of $R_1 \vee \ldots \vee R_n$ in D'_K are closed. Since $R_1 \vee \ldots \vee R_n$ may be used in D_R several times, the result is a refutation D''_K of $M \cup \{R_1 \vee \ldots \vee R_n\}$ with top clause K. In order to turn D''_K into a refutation of M first replace in D''_K every occurence of the clause $R_1 \vee \ldots \vee R_n$ by D. This results in a derivation D'''_K of several occurences of the clause L from M. In order to turn this into a refutation, append D'''_K with D_L as many times until all occurences of L are closed. Since D_L is in the above normal form, the clause L is no longer used in this final derivation. Thus we obtain the desired refutation of M. **Q.E.D.**

6. Conclusions

We have developed a partial and a total variant of the model elimination calculus for theory reasoning and proved their completeness. For this purpose we gave a sufficient completeness criterion for the theory reasoning system, but left open the question how such a system can be obtained from given theory axioms. This is currently being investigated (Bau92b). Finally I would like to thank U. Furbach for reading an earlier draft of this paper.

References

(AB70) R. Anderson and W. Bledsoe. A linear format for resolution with merging and a new technique for establishing completeness. *J. of the ACM*, 17:525–534, 1970.

(Bau91) P. Baumgartner. A Model Elimination Calculus with Built-in Theories. Fachbericht Informatik 7/91, Universität Koblenz, 1991.

(Bau92a) P. Baumgartner. An Ordered Theory Resolution Calculus. In *Proc. LPAR '92*, 1992. (To appear).

(Bau92b) P. Baumgartner. Completion for Linear Deductions. (in preparation), 1992.

(BF92) P. Baumgartner and U. Furbach. Consolution as a Framework for Comparing Calculi. (in preparation), 1992.

(BGL85) R. Brachman, V. Gilbert, and H. Levesque. An Essential Hybrid Reasoning System: Knowledge and Symbol Level Accounts of Krypton. In *Proc. IJCAI*, 1985.

(Bib87) W. Bibel. *Automated Theorem Proving.* Vieweg, 2nd edition, 1987.

(CL73) C. Chang and R. Lee. *Symbolic Logic and Mechanical Theorem Proving.* Academic Press, 1973.

(FHS89) Ulrich Furbach, Steffen Hölldobler, and Joachim Schreiber. Horn equational theories and paramodulation. *Journal of Automated Reasoning*, 3:309–337, 1989.

(Fit90) M. Fitting. *First Order Logic and Automated Theorem Proving.* Texts and Monographs in Computer Science. Springer, 1990.

(GNPS90) J. Gallier, P. Narendran, D. Plaisted, and W. Snyder. Rigid E-unification: NP-Completeness and Applications to Equational Matings. *Information and Computation*, pages 129–195, 1990.

(Lov78a) D. Loveland. *Automated Theorem Proving - A Logical Basis.* North Holland, 1978.

(Lov78b) D. W. Loveland. Mechanical Theorem Proving by Model Elimination. *JACM*, 15(2), 1978.

(LSBB92) R. Letz, J. Schumann, S. Bayerl, and W. Bibel. SETHEO: A High¡-Performace Theorem Prover. Journal of Automated Reasoning, 1992.

(MR87) N. Murray and E. Rosenthal. Theory Links: Applications to Automated Theorem Proving. *J. of Symbolic Computation*, 4:173–190, 1987.

(Ohl86) Hans Jürgen Ohlbach. The Semantic Clause Graph Procedure – A First Overview. In *Proc GWAI '86*, pages 218–229. Springer, 1986. Informatik Fachberichte 124.

(Ohl87) Hans Jürgen Ohlbach. Link Inheritance in Abstract Clause Graphs. *Journal of Automated Reasoning*, 3(1):1–34, 1987.

(Pet90) U. Petermann. Towards a connection procedure with built in theories. In *JELIA 90.* European Workshop on Logic in AI, Springer, LNCS, 1990.

(Sie89) Jörg H. Siekmann. Unification Theory. *Journal of Symbolic Computation*, 7(1):207–274, January 1989.

(Sti83) M.E. Stickel. Theory Resolution: Building in Nonequational Theories. SRI International Research Report Technical Note 286, Artificial Intelligence Center, 1983.

(Sti85) M. E. Stickel. Automated deduction by theory resolution. *Journal of Automated Reasoning*, pages 333–356, 1985.