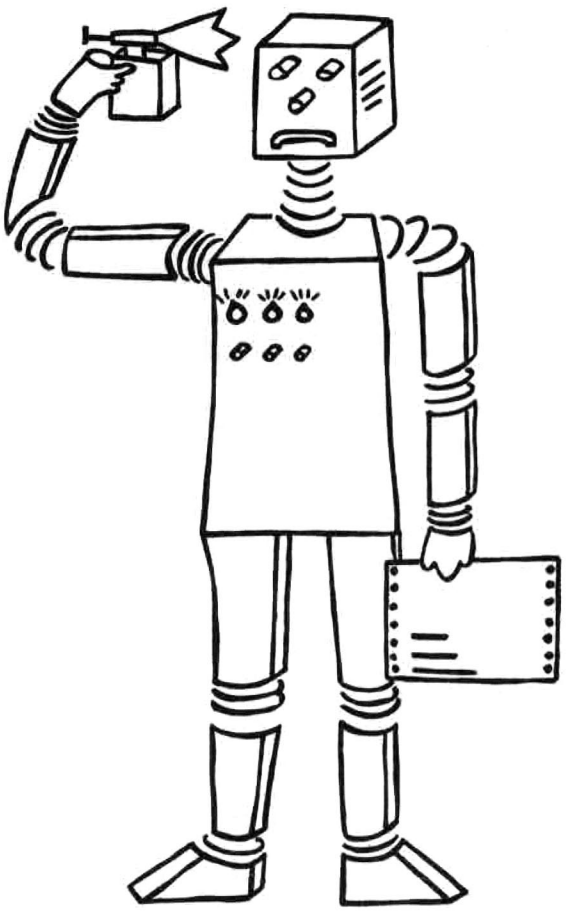
SEKI-REPORT

Artificial
Intelligence
Laboratories

Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern 1, W. Germany



SOME UNDECIDABLE CLASSES OF CLAUSE SETS

Manfred Schmidt-Schauß

Juni 1986 SEKI-REPORT SR-86-08

Some Undecidable Classes of Clause Sets.

Schmidt-Schauss, Manfred; Universität Kaiserslautern; 6750 Kaiserslautern, F.R. Germany
(UUCP: seismo!mcvax!unido!uklirb!schauss

This work is supported by the Deutsche Forschungsgemeinschaft, SFB 314

Abstract.

The undecidability of finitely generated stable transitive relations on free terms is proved in an elementary way and then used to show the undecidability of the implication $A \Rightarrow B$ of two clauses A and B. The implication is decidable in the case A has at most two literals that are complementary unifiable after renaming. Application of the undecidability of transitive relations yields more classes of clause sets that have an undecidable satisfiability problem: clause sets consisting of two clauses with two literals each (2-clauses) and in addition 2 ground units, clause sets consisting of a single 3-clause and arbitrarily many (non-ground) units and clause sets consisting of a 4-clause and arbitrarily many ground units. Finally we show the undecidability of the so-called D-clause sets, where a D-clause is a Horn-clause of the form $Q_1(x_1) \wedge \ldots \wedge Q_n(x_n) \Rightarrow Q_{n+1}(t)$.

Keywords: Undecidability, Clause sets, Implication of Clauses, Declarations,

1. Introduction.

We are interested in problems in Automated Deduction and Logic Programming, where the following notation is standard: A clause is a disjunction of literals (represented as a set of literals). A clause with n literals is called n-clause. A literal is an atom or its negation and an atom is of the form $P(t_1,\ldots,t_n)$, where P is the predicate letter and $t_1,\ldots,t_n$ are first order terms constructed in the usual way out of variable, constants and n-ary function symbols. T is the set of terms. A literal is ground , iff it contains no variables. A subtitution σ is a finite endomorphism on the set of terms that can be represented as a set of variable term pairs $\{x_1 \leftarrow t_1 , \ldots , x_n \leftarrow t_n\}$. Σ is the set of all substitutions.
We say a class of clause sets is (un)decidable, if we mean the satisfiability of the clause set is (un)decidable, in other words that there does (not) exist an algorithm for deciding the satisfiability.
Our notation is consistent with the conventions in automated theorem proving]Lo78] and unification theory [Si84].
We are particularily interested in the following problems:

i) Dixon´s Z-resolution [Di73, Oh83], which poses the problem of finding a unification procedure with built-in 2-clauses.
ii) Removing implied clauses from clause sets or knowledge bases consisting of clauses [Gt86], which is a generalization of removing subsumed clauses from clause sets [Ei81] This requires to solve the problem, whether a clause A implies a clause B.
iii) Unification in a sorted signature, where a finite number of sort-declarations is used to construct the sort of a term [Sch85].
iv) The explicit treatment of recursive clauses in PROLOG [CM81] or automated theorem provers for Horn-clauses, i.e. the problem of the compilation of special classes of recursive clauses into a terminating, nonrecursive algorithm.

The undecidability of each of these problems is a consequence of the result proved in the first section:
we show that in a finitely generated transitive stable relation on free terms, it is not decidable whether two terms are in this relation. The proof is based on the halting problem for register machines with at least two registers.
This result is related to the undecidability of first order equational logic, but it is more handy.
There are numerous results on pure first order formulas (i.e. first order formulas without constants and functions) with a fixed quantor-prefix (see for example [DG79,Le79, Go84] and [Jo73] for decision procedures for such classes). Unfortunately, not all of these results are directly translatable into results on classes of clause sets that obey some syntactic restriction, since the normal first order predicate calculus allows arbitrary functions and constants. For some of these translatable results we give alternative, shorter proofs.

2. Transitive Relations are Undecidable.

We consider transitive, stable and finitely generated relations $<$ on **T**, the set of terms.That is we are interested in relations $<$ for which the following three conditions are satisfied:

2.1 Definition

1) $<$ is transitive, i.e. $s < r$ and $r < t$ implies $s < t$.
2) $<$ is generated by a finite set of relations $s_i < t_i$, $1 \le i \le n$.
3) $<$ is stable, i.e. $\forall\sigma \in \Sigma; \forall s,t \in T: \; s < t \Rightarrow \sigma s < \sigma t$.

We call such a relation a TS-relation.

A standard definition of a register machine [EF78] is:

2.2 Definition. A register machine uses registers $R_1,...,R_m$ and an alphabet $A = \{a_1,...,a_n\}$. Program lines have the following format:

1) Z LET $R_i = R_i + a_j$ (Z is the line number)
   "the character a_j is pushed onto the register stack R_i."
2) Z LET $R_i = R_i - a_j$
   "If the last character in R_i is a_j, then delete it"
3) Z IF R_i = e THEN Z´ ELSE Z_1 OR ... OR Z_n. (e denotes the empty stack)
   "If R_i is empty, go to Z´ else if the last character in R_i is a_j go to Z_j.
4) Z PRINT
5) Z STOP ■

For the class of register machines with at least two registers it is well-known that the halting problem is undecidable [EF78], that means, there is no algorithm which accepts the program of a register machine and prints "1" whenever the register program stops starting with empty registers and prints "0" otherwise.
Furthermore, there exists a register program such that there is no algorithm which answers correctly the question whether this program stops for a given input:

2.2 Theorem.

i) The halting problem is undecidable for register machines with at least 2 registers starting with the empty registers.
ii) The halting problem is undecidable for some fixed register machines with at least 2 registers and arbitrary input.

We describe how a register program RM can be translated into a TS-relation $<_{RM}$. This translation and the following undecidability results are a technical prerequisite for the the results in section 3.:
Let a program for a register machine be given. We use a signature that includes as constants the alphabet $A = \{a_1,...,a_n\}$ of the register machine and enough natural numbers to cover all the line numbers. REG is a binary function, which denotes the contents of one register $(b_1...b_k)$ in the form $REG(REG(...(REG(e,b_1),..),b_{k-1}),b_k)$. The empty register is represented as e. ST is a function with m+1 arguments, which corresponds to the status of execution. The first argument is the line number and the next m arguments are for the register contents.
Every program line corresponds to one or more relations of the form $ST(Z_1...) <_{RM} ST(Z_2...)$, which are used as generators for the relation. For technical reasons, we assume that the program has a unique START-relation with Z=1 and that all other program lines are enumerated starting with Z=2. A program execution that stops corresponds to the fact that the left term of the START line and the right term of the STOP line are in the relation $<_{RM}$.

If a program statement is encoded into several relations, then the partitioning must be unique to guarantee a unique program execution.

We go through the possibilities for program lines and describe their coding:
The START-statement is encoded as $ST(1, x_1,...,x_m) <_{RM} ST(2,x_1,...,x_m)$.
We assume Z > 1 for all other statements.

1) Z LET $R_i = R_i + a_j$ is encoded in one relation:
   $ST(Z, x_1,...,x_m) <_{RM} ST(Z+1, x_1,...,REG(x_i, a_j),...,x_m)$,

(where Z+1 is a constant, not a term)

2) Z LET $R_i = R_i - a_j$ is encoded in n+1 relations (in each case R_i corresponds to the $(i+1)^{th}$ argument.

$ST(Z, x_1,...,e,...,x_m) <_{RM} ST(Z+1, x_1,...,e,...x_m))$.

For $k \neq j$:

$ST(Z, x_1,...,REG(x_i,a_k),...,x_m) <_{RM} ST(Z+1, x_1,...,REG(x_i,a_k),...,x_m)$

In case the last character of R_i is a_j:

$ST(Z, x_1,...,REG(x_i,a_j),...,x_m) <_{RM} ST(Z+1, x_1,...,x_i,...x_m)$

3) Z IF R_i = e THEN Z´ ELSE Z_1 OR ... OR Z_n. is encoded in n+1 relations:

For the empty register R_i:

$ST(Z, x_1,...,e,...,x_m) <_{RM} ST(Z´, x_1,...,e,...x_m)$

For j = 1,...,n:

$ST(Z, x_1,...,REG(x_i,a_j),...,x_m) <_{RM} ST(Z_j, x_1,..., REG(x_i,a_j),...,x_m)$

4) Z PRINT. is not relevant for the halting problem.

5) Z STOP. is encoded in one relation:

$ST(Z,x_1,...,x_n) <_{RM} ST(Z_{STOP},y_1,...,y_n))$.

Z_{STOP} is a line number not corresponding to a program line. We can assume that there is only one such STOP-line. Note that the registers are not transferred.

Example. Assume we have reached the program status $ST(Z,t_1,...,t_{i-1},REG(t_i,a_j),t_{i+1},...,t_n)$. The only possibility to find a greater status w.r.t. $<_{RM}$ is to look for the line with number Z. Assume line Z is of form 2). and the term is unifiable with the left term of the relation in case iii). Then the next reachable status is $ST(Z+1,t_1,...,t_i,...,t_n)$. ■

Now Theorem 2.3 implies the desired results on TS-relations:

2.4 Theorem . Let < be a TS-relation.

i) For all ground terms s,t s < t is undecidable for the class of all TS-relations.,

ii) There exists a TS-relation <, such that s < t for arbitrary ground terms s and t is undecidable.

iii) There exists a TS-relation < and a ground term t such that s < t for an arbitrary ground term s is undecidable.

Proof. Let the TS-relation $<_{RM}$ be generated from a register machine as described above.

i) We choose s = ST(1,e,...,e) and t = $ST(Z_{STOP},e,...,e)$. Then the decidability of $s <_{RM} t$ for all relations would imply the decidability of the halting problem for register machines.

ii) and iii): Fix $<_{RM}$ to be the TS-relation corresponding to the existing program in Theorem 2.3 ii). Then choose s = $ST(1,s_{g,1},...,s_{g,n})$ and t = $ST(Z_{STOP},e,...,e)$ where $s_{g,i}$ are arbitrary ground terms representing the register contents. Now Theorem 2.2 ii) is applicable.

2.5 Definition. A regular unique TS-relation is defined as:

i) For all generating relations $s_i < t_i$: $V(s_i) = V(t_i)$.

ii) For all ground terms s,t with s < t: If t is a direct successor of s, then t is unique. ■

The construction in the proof of the previous theorem yields the following lemma

2.6 Lemma. The undecidability results in Theorem 2.4 are true even if for regullar unique TS-relations.

Proof: We can extend the program of a register machine after the STOP with a group of statements, which empties all registers and then stops, which shows the result. ■

3. Some Undecidable Clause Sets.

Our main interest is the analysis of the implication problem $A \Rightarrow B$, i.e. whether or not the implication $A \Rightarrow B$ where A and B are clauses is decidable.

In automated theorom proving systems as well as in knowledge bases where clauses are used as facts and rules it is desirable to remove redundant clauses from clause sets (from the knowledge base). In automated theorem provers the (decidable) subsumption rule is used as redundancy deletion rule. The deletion of implied clauses is a generalization of the deletion of subsumed clauses and may be more powerful.

However, in this section it is shown that the implication of two clauses is in general undecidable. For knowledge bases where all clauses are Horn-clauses the problem remains open.

If a clause A implies a clause B, then A and $\neg B$ is contradictory (and vice versa) hence the implication problem is equivalent to the problem whether or not the class of clause sets consisting of one clause A and ground units B_i is decidable.

In this section we consider clause sets consisting of one clause and ground units and some related clause sets. Furthermore we vary the length of the clause A.

The following result appears to be well-known [Go86], however, we give a proof that provides an efficient decision procedure for the clause sets in question.

3.1 Proposition. The class of clause sets consisting of a 2-clause and some ground clauses is decidable.

Proof. The nontrivial case is that L and M are complementary unifiable.

Let {L,M} be the 2-clause. An algorithm for deciding the unsatisfiability works as follows:

compute the n-fold self-resolvent $R_n := \{L_n, M_n\}$ of {L, M}. R_{n+1} can be computed in two ways: i) Let θ be the most general unifier of M_n and L. Then $R_{n+1} := \{\theta L_n, \theta M\}$

ii) Let μ be the most general unifier of L_n and M. Then $R_{n+1}' := \{\mu L, \mu M_n\}$.

Since up to renaming of variables R_{n+1} is unique, there exist substitutions σ_n and τ_n, such that $L_{n+1} = \sigma_n L_n$ and $M_{n+1} = \tau_n M_n$. Hence either the depth of terms in R_n grows indefinitely or the process of self-resolving does not generate new resolvents.If the depth of terms in R_m exceeds the maximum of the term depth in the ground units, then for all $m' \geq m$, one literal in $R_{m'}$ is not resolvable with a ground unit. ■

Remark. The decidability of the class of clause sets consisting of one 2-clause and two arbitrary units is an open problem. As W. Goldfarb mentioned [Go86], H.R. Lewis has worked on this problem without success.

The following theorem follows from a known result on formulas with a Krom-matrix with at most 2 literals in every disjunct [DG79,Le79]. We give a proof as an application of Theorem 2.4

3.2 Theorem.

The class of clause sets consisting of some 2-clauses and two ground unit clauses L and M is undecidable

Proof: Let $<$ be a TS-relation generated by $s_i < t_i$ and let P be a unary predicate. Let the clause set be $CS := \{\{\neg P(s_i), P(t_i)\} \mid 1 \le i \le n\}$. Then we have:

$s < t \Leftrightarrow (CS \cup \{P(s), \neg P(t)\}$ is contradictory).

Now the result follows from Theorem 2.4 ■

Goldfarb (cf. [Go74]) has shown that the class of clause sets with two 2-clauses is undecidable.
The corresponding clause set is of the form {{P, P}, {-P, -P}}
Lewis Le79] has proved the undecidability of formulas with at most 6 atomic formulas, which form in fact a clause set with two 2-clauses and two (nonground) units.
We give an elementary proof for the undecidability of a Horn-clause set with two 2-clauses and two ground units as an application of the undecidability of TS-relations.

3.3 Theorem. The class of Horn-clauses consisting of two 2-clauses and two ground units L and M is undecidable.

Proof: We show that this problem is equivalent to the undecidability of TS-relations:

Let $s_i < t_i$ be generating relations for $<$ and let s,t be ground terms.We can assume that $s \neq t$ and that $<$ is a regular unique TS-relation. Let P be a binary predicate, f be a new ternary function and c_i be new constants. Let the clause set CS be:

i) $P(x\ y) \Rightarrow P(x\ f(y_1\ y_2\ y))$

ii) $P(x\ f(s_1\ t_1\ f(s_2\ t_2\ f(\ldots\ f(s_n\ t_n\ c_1)\ \ldots)) \Rightarrow P(y_1\ f(x\ y_1\ y_2))$
where $V(s_i,t_i) \cap V(s_j,t_j) = \emptyset$ for $i \neq j$ and y_1 and y_2 are new variables.

iii) $P(s\ r_g)$, where r_g is a ground instance of $f(s_1\ t_1\ f(s_2\ t_2\ f(\ldots\ f(s_n\ t_n\ c_1)\ \ldots)$

iv) $\neg P(c_2\ f(t\ c_2\ c_3))$

Now we have to show that: CS unsatisfiable $\Leftrightarrow$ $s < t$.

"$\Leftarrow$": Let $s < t$. Then there exists a chain of terms q_j such that $s = q_0 < q_1 < \ldots < q_k = t$ and the relations in the chain are instances of the generating relations.

We show, that there is a refutation for CS:

1) We resolve iii) with the first literal in ii) and obtain the unit $P(y_1\ f(s\ y_1\ y_2))$.
2) Let $q_0 < q_1$ be an instance of $s_j < t_j$. Then resolving the unit obtained in 1) with i) j-1 times we obtain a new unit of the form $P(z_1\ f(x_1\ y_1\ f(x_2\ y_2\ f(\ldots\ f(x_{j-1}\ y_{j-1}\ f(s\ z_1\ z_2)\ \ldots)$.
3) resolving this new unit with ii) we obtain the unit $P(y_1\ f(\sigma t_j\ y_1\ y_2))$, where σ is the most general unifier of s and s_j.
4) We can go along the chain until we reach the unit $P(y_1\ f(t\ y_1\ y_2))$, which is complementary unifiable with the unit in iv)

"$\Rightarrow$": We show that a refutation can only be obtained in the same way as above:

Since the clause set consists of Horn-clauses, an input resolution proof is possible.

1) The unit in iii) is resolvable with i) and ii). It is easy to see, however, that resolution with i) cannot contribute to a contradiction, since the produced units are only resolvable with i). Thus the only sensible unit is $P(y_3\ f(s\ y_3\ y_4))$.
2) This unit is not resolvable with iv) since $s \neq t$. It is resolvable with i) and possibly with ii).Resolution with i) gives infinitely many units, from which at most one is resolvable with the first literal in ii) due to Lemma 2.6. If none of these deduced units is resolvable with i), the clause set is satisfiable.Hence we obtain a unit $P(y_1\ f(q_1\ y_1\ y_2))$
3) Using the unit obtained from resolving against ii), we can construct step by step a

<-chain.Since CS is unsatisfiable, a proof exists, hence we reach the unit in iv) after a finite number of steps. ■

Now Theorem 2.4 implies the result.

3.4 Corrollary. Let < be a TS-relation on terms.

i) If < is generated by one relation, then s < t is decidable for ground terms.

ii) There is no algorithm that decides for all relations generated by 2 relations the relations s < t for ground terms. ■

3.5 Proposition: A clause set consisting of a 3-clause and some nonground units is undecidable:

Proof: Let the 3-clause be the transitivity clause: $\{\neg P(x\ y), \neg P(y\ z), P(x\ z)\}$ and let the units be $P(s_i\ t_i)$ and $\neg P(s\ t)$. Now theorem 2.4 i) implies the desired conclusion. ■

Goldfarb [Go86] mentioned that Lewis has proved the stronger theorem that a clause set with a 3-clause and three units is undecidable. However, the proof is not published.

Now we are able to show that the problem whether a clause A implies a clause B is undecidable. This problem is equivalent to the decidability of a clause set consisting of the clause A and some ground units. We show that the implication remains undecidable if A is an n-clause with $n \leq 4$.

3.6 Theorem.

i) The class of clause sets consisting of one clause and ground unit clauses L_i is undecidable.

ii) The class of clause sets consisting of one clause with 4 literals and ground unit clauses L_i is undecidable.

Proof. It suffices to show part ii).

We show that the problem of satisfiability of two 2-clauses and some ground units can be encoded (see Theorem 3.3): Without loss of generality we can assume that the clause set CS1 consists of the following clauses:

CS1: $\{\neg P(s_1\ h(c)), P(t_1\ h(c))\}$, $\{\neg P(s_2\ h(c)), P(t_2\ h(c))\}$, $P(s\ h(c))$, $\neg P(t\ h(c))$

c is a constant, h is a unary function not occurring elsewhere and s and t are ground terms.

Let the clause set CS2 be:

CS2: $A := \{\neg P(s_1\ x_1), P(t_1\ x_1), \neg P(s_2\ h(x_1), P(t_2\ h(x_1))\}$

and the following ground units:

$P(\sigma_{g1} s_1\ , c)$, $\neg P(\sigma_{g1} t_1\ , c)$, $P(\sigma_{g2} s_2\ , h(h(c)))$, $\neg P(\sigma_{g2} t_2\ , h(h(c))$; and $P(s\ h(c))$, $\neg P(t\ h(c))$

We can assume, that $V(s_1, t_1) \cap V(s_2, t_2) = \emptyset$ and x_1 does not occur elsewhere.

We have to show that: CS1 is satisfiable $\Leftrightarrow$ CS2 is satisfiable.

"$\Leftarrow$": Obviously the 2 2-clauses are deducable from the 4-clause and the ground units.

"$\Rightarrow$": Assume we have a model M of CS1. M consists of a maximal set of ground literals. The relevant part of M are the literals with h(c) as second argument. We can assume that M contains the 4 additional ground units of CS2. We change M in the following way to obtain a model of CS2: We assume that all the ground units $P(t_{2,g}\ r_g)$ are in M, where $t_{2,g}$ is a ground instance of t_2 and r_g is a ground term, but not equal to c, h(c) or h(h(c)). There are no conflicts with the relevant units which are in M. We show that the

changed M is a model of CS2: Therefore it suffices to show that all ground instances of A are valid in M.

If a ground substitution replaces x_1 by c, then one of the third and fourth literal is true. If a ground substitution replaces x_1 by h(c), then the first or second literal is true. In all other cases, the literal $P(t_{2,g}\, r_g)$ is true. The changed M is a model of the 4-clause and all ground units.

Now Theorem 3.3 is applicable. ■

3.7 Corollary. Let A be a clause. Then

i) $A \Rightarrow B$ is decidable provided A is a 2-clause.

ii) $A \Rightarrow B$ is undecidable if A is a clause with four or more literals. ■

The decidability of $A \Rightarrow B$ where A a 3-clause is an open question. The next theorem gives a sufficient criterion for the decidability of the implication problem. More sufficient criterions for the decidability are given in [Gt85].

We give a criterion for recognizing decidable subcases of the implication problem:

3.8 Theorem. Let A be a clause, such that there are at most two literals in A that are complementary unifiable after renaming. Then it is decidable, whether A implies an arbitrary clause B.

Proof. The arguments are similar to those in the proof of 3.1: Let CS be a clause set satisfying the preconditions of this theorem. Let CS consist of the clause A and some ground units.If the clause set is unsatisfiable, a contradiction, i.e. the empty clause, can be found first producing self-resolvents of A and then resolving with the ground units. We argue that a sligthly modified procedure has the same property and recognizes the satisfiability of the clause set:

Let the clause A consist of the complementary unifiable literals K and L and the rest M.Let m be the number of ground literals resolvable with literals in M. Every clause C deduced from A by resolving and instantiating has two parts: The KL-part consisting of two instances of K and L and the M-part consisting of some instances of literals in M. The following set of instances of deduced clauses is important:

I(C): = {I| I is an instance of C, such that only the variables occurring in the M-part of C are instantiated. The M-part of I has at most m literals, every literal in M is resolvable with some ground unit in CS}

The procedure works as follows:

1) Let IS_0 = I(A).
2) DO until IS_n does not change any more:
   $IS_{n+1} := IS_n$
   Resolve all clauses in IS_n with IS_0 in all possible ways .
   Add I(R) to IS_{n+1} for all such resolvents.
   Remove clauses with literals in the KL-part that are not unifiable with a ground literal.
   Remove equivalent clauses from IS_{n+1}.
   n := n+1.
3) Resolve the clauses in IS_n with the ground units in all possible ways.
   If the empty clause is produced, then the clause set is unsatisfiable, otherwise satisfiable.

This procedure terminates, since the clauses in IS_n are bounded in depth and length.

It remains to show, that the empty clause can always be reached for unsatisfiable clause sets.
Let RS_n be the set of n-fold self-resolvents of A. Let $RSI_n := \cup \{I(R) | R \in RS_n\}$.
To complete the proof, it suffices to show, that $RSI_n = IS_n$ for all n. Obviously we have $RSI_n \supseteq IS_n$. We show $RSI_n \subseteq IS_n$ by induction.
Let $R_1 \in RS_n$ consist of two literals K_1, L_1 and of a rest M_1 and let $R_2 \in IS_0$ consist of two literals K_2, L_2 and of a rest M_2. A resolvent of R_1 and R_2 is $R_3 := \{\sigma L_1, \sigma K_2, \sigma M_1, \sigma M_2\}$ where σ is a most general unifier of K_1 and L_2 .Let $R_4 := \theta R_3$ be in RSI_n and let $(\theta\sigma)|_M$ be $\theta\sigma$ restricted to the variables in M_1 and M_2. Then we have $(\theta\sigma)|_M R_2 \in IS_n$ by the induction hypothesis and $(\theta\sigma)|_M R_1 \in IS_0$ (up to renaming). The resolvent of $(\theta\sigma)|_M R_1$ and $(\theta\sigma)|_M R_2$ is equivalent to R_4, hence $R_4 \in IS_{n+1}$. ■

Remark: The remaining open questions are:

i) Is the class of clause sets consisting of one 2-clause and arbitrary units decidable?
ii) Is the class of clause sets consisting of one 3-clause and ground units decidable?
iii) Is the class of clause sets consisting of one Horn-clause and ground units decidable?

The hard part of ii) is a special case iii). In other words, it is not clear whether or not the implication $A \Rightarrow B$ is decidable, where A is a Horn-clause.

Undecidability of D-Clause Sets.

In this paragraph we consider the problem of decidability of unification of sorted terms, where the sort of a terms is determined by a finite set of declarations [GM85,Gg83,SS85] . I.e. there is given a finite partially ordered set of sorts and a finite set of declarations t:S (the term t is of sort S). The sort of a term t' is of sort less than S, if t:S is a declaration and t' is an instance of t. The corresponding (well-sorted) substitutions substitute for every variable x a term of lesser sort.

The problem of the decidability of unification of sorted terms in this case is equivalent to the decidability of a special class of Horn clauses. We give here a proof for the undecidability of this class of clause sets.

We define a special subclass of sets of Horn clauses, which are called D-clauses (D for declarations):

4.1 Definition. Let $P_1, \ldots, P_m$ be unary predicates.

Let CS be a set of Horn-clauses, such that every axiom-clause $C \in CS$ is of type $A \Rightarrow B$

1) A may be empty;
   if A is not empty, then A is of the form $Q_1(x_1) \wedge \ldots \wedge Q_n(x_n)$, where Q_i is one of the predicates P_i and the x_i are distinct variables.
2) B is of the form $Q_i(t)$, where t is a term. Furthermore $V(A) \subseteq V(t)$.

The theorem clause C_T is of the same form, except that B is negated. C_T is the negated clause of
$\exists s: Q_1(x_1) \wedge \ldots \wedge Q_n(x_n) \wedge Q_{n+1}(s)$,where x_i are variables in s.

4.2 Theorem. D-clause sets are undecidable.

Proof.

Let $s_i < t_i$, $1 \le i \le n$ be the generating relations of <, satisfying theorem 2.3 ii).
Let f be a binary function symbol not contained in any term s_i, t_i.
Let the D-clause set DCS be:
D_0: $P(f(t, x))$

D_i : $\{\neg P(y)\ ,\ P(f(s_i\ f(t_i\ y))\}$; for $1 \le i \le n$.

Let the theorem clause be: $\{\neg P(z), \neg P(f(s,z))\}$.

We show that $s < t \Leftrightarrow$ DCS is unsatisfiable:

"$\Rightarrow$": $s < t$ implies that there exist a chain q_j with $s = q_0 < q_1 < \ldots < q_k = t$ and every relation in the chain is an instance of a generating relation. Starting the refutation with the second literal of the theorem clause, we obtain succesively the clauses $\{\neg P(z), \neg P(f(q_i,z))\}$ and finally $\{\neg P(z), \neg P(f(t,z))\}$. resolution with D_0 yields the empty clause.

"$\Leftarrow$": Assume DCS is unsatisfiable. Then a proof can be found by resolving D_0 and D_1 until two units $P(r_1)$ and $P(f(s_j\ r_2)$ are deduced such that these units are compatibly unifiable with the theorem clause.That means r_1 and r_2 are unifiable. All deduced units are of the form $P(f(s_i\ f(t_i\ f(s_j\ f(t_j \ldots f(t\ x) \ldots)$. The instances of two such clauses directly gives a chain $s = q_0 < q_1 < \ldots < q_k = t$. ■

Now Theorem 2.4 is applicable.

References.

Bo85 Börger, E., Berechenbarkeit, Komplexität , Logik ,Vieweg &Sohn, Braunschweig/Wiesbaden (1985)

CM81 Clocksin, W.F.,Mellish,C.S., Programming in PROLOG, Springer-Verlag, (1981)

Di73 Dixon, J., Z-resolution: theorem proving with compiled axioms, JACM 20,1 (1973)

DG79 Dreben, B and Goldfarb, W., The decision problem: Solvable classes of quantificational formulas, Addison-Wesley, Reading, Massachusetts,(1979)

EF78 Ebbinghaus, H.-D., Flum, J., Thomas, W., Einführung in die mathematische Logik, Wissenschaftliche Buchgesellschaft, Darmstadt (1978)

Ei81 Eisinger, N., Subsumption and Connection Graphs, Proc. IJCAI 1981, Vancouver,

GM84 Goguen, J.A., Meseguer, J., Order Sorted Algebra I. Partial and overloaded operators, Error and Inheritance. SRI-report (1985)

Gg83 Gogolla, M., Algebraic specifications with partially ordered sorts and declarations. Techn. Report, Institut für Informatik, Dortmund (1983)

Go74 Goldfarb, W., On decision problems for quantification theory , Ph. D. Thesis, Harvard University, (1974)

Go84 Goldfarb,W., The unsolvability of the Gödel class with identitiy, Journal of Symbolic Logic 49 ,4, pp. 1237-1252, (1984)

Go86 Goldfarb, W., private communication. (1986)

Gt86 Gottlob, G., Subsumption and Implication, to appear in Information Processing letters

Jo73 Joyner, W. H., Jr., Automatic Theorem-proving and the decision problem, Report #7-73, Center for research in computing technology, Harvard University, Cambridge Massachusetts (1973)

Le79 Lewis, H. R., Unsolvable classes of quantificational formulas, Addison-Wesley, Advanced Book Program, Reading, Massachusetts, (1979)

LG73 Lewis, H.R., Goldfarb, W., The decision problem for formulas with a small number of atomic subformulas, Journal of Symbolic Logic 38 (3), pp. 471 -480, (1973)

Lo78 Loveland, D., Automated Theorem Proving: A logical Basis, Fundamental Studies in Computer Science, North Holland,(1978)

Oh83 Ohlbach, H.J., Ein Regelbasiertes Klauselgraph-Beweisverfahren, 7th German Workshop on Artificial Intelligence, Informatik-Fachberichte 76, (ed. Bernd Neumann), Springer-Verlag (1983)

Si84 Siekmann, J.H., Universal Unification, Proc. of the 7th CADE, (ed. R.E. Shostack), Springer-Verlag LNCS 170, pp. 1-42, (1984)

SS85 Schmidt-Schauss, M., Unification in a Many-sorted Calculus with Declarations, 9th German Workshop on Artificial Intelligence, Springer-Verlag (1985) (to appear)

Ta75 Taylor, W.,Equational Logic, Proc. of the Colloq. held in Szeged, Coll. Math. Soc. János Bolyai, vol 17, pp. 465-501, North-Holland (1975)