Automated Deduction by Theory Resolution *

Mark E. Stickel

Artificial Intelligence Center

SRI International

and

Center for the Study of Language and Information

Stanford University

May 14, 1985

*This research was supported by the Defense Advanced Research Projects Agency under Contract N00039-84-K-0078 with the Naval Electronic Systems Command and was also made possible in part by a gift from the System Development Foundation. The views and conclusions contained in this document are those of the author and should not be interpreted as representative of the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States government.

Abstract

Theory resolution constitutes a set of complete procedures for incorporating theories into a resolution theorem-proving program, thereby making it unnecessary to resolve directly upon axioms of the theory. This can greatly reduce the length of proofs and the size of the search space. Theory resolution effects a beneficial division of labor, improving the performance of the theorem prover and increasing the applicability of the specialized reasoning procedures. Total theory resolution utilizes a decision procedure that is capable of determining unsatisfiability of any set of clauses using predicates in the theory. Partial theory resolution employs a weaker decision procedure that can determine potential unsatisfiability of sets of literals. Applications include the building in of both mathematical and special decision procedures, e.g., for the taxonomic information furnished by a knowledge representation system. Theory resolution is a generalization of numerous previously known resolution refinements. Its power is demonstrated by comparing solutions of "Schubert's Steamroller" challenge problem with and without building in axioms through theory resolution.

1 Introduction

Incorporating a theory into derived inference rules so that its axioms are never resolved upon has enormous potential for reducing the size of the exponential search space commonly encountered in resolution theorem proving [9, 18, 4, 35]. Theory resolution is a method of incorporating specialized reasoning procedures in a resolution theorem prover so that the reasoning task will be effectively divided into two parts: special cases, such as reasoning about inequalities or about taxonomic information, are handled efficiently by specialized reasoning procedures, while more general reasoning is handled by resolution. The connection between the two reasoning components is made by having the resolution procedure resolve on sets of literals whose conjunction is determined to be unsatisfiable by the specialized reasoning procedure, rather than by just using ordinary unification of complementary literals. The objective of research on theory resolution is the conceptual design of deduction systems that combine deductive specialists within the common framework of a resolution theorem prover.

We are incorporating our work on theory resolution in the development of a deduction system [30] for use in the KLAUS natural-language-understanding system [15]. The use of theory resolution to help incorporate theories of taxonomies, orderings, etc., should be very beneficial, given the pervasive need for these forms of reasoning in carrying out the task of understanding language and the real world.

Concern has often been expressed about the ineffectiveness of applying resolution theorem proving to problems in artificial intelligence. Theory resolution is designed to partly address this concern by providing a means for incorporating specialized reasoning procedures in a resolution theorem prover. The division of labor achieved in the reasoning process by theory resolution is intended to produce the dual advantages of improving the theorem prover's performance by the use of more efficient reasoning procedures for special cases and of increasing the range of application of the specialized reasoning procedures by including them in a more general reasoning system.

Past criticisms of resolution can often be characterized by their pejorative use of the terms *uniform* and *syntactic*. Theory resolution meets these objections head-on. In theory resolution, a specialized reasoning procedure may be substituted for ordinary syntactic unification to determine unsatisfiability of sets of literals. Because the implementation of this specialized reasoning procedure is unspecified—to the theorem prover it is a "black box" with prescribed behavior, namely, able to determine unsatisfiability in the theory it implements—the resulting system is nonuniform because reasoning within the theory is performed by the specialized reasoning procedure, while reasoning outside the theory is performed by resolution. Theory resolution can also be regarded as being not wholly syntactic, since the conditions for resolving on a set of literals are no longer based on their being made syntactically identical, but rather on their being unsatisfiable in a theory, and thus resolvability is partly semantic.

Besides being a higher-level inference rule than ordinary resolution by virtue of its use of specialized reasoning procedures, theory resolution is also a higher-level rule because, as in hyperresolution, it allows inferences requiring more than two parent clauses.

Theory resolution can be seen partly as an extension of the work on building in equational theories (i.e., theories that can be expressed as a set of equalities) [25]. This consisted of using special unification algorithms and reducing terms to normal form. This work has been extended substantially, particularly in the area of development of special unification algorithms for various equational theories [28].

Not all theories that it would be useful to incorporate are equational. For example, reasoning about orderings and other transitive relations is often necessary, but using ordinary resolution for this is quite inefficient. It is possible to derive an infinite number of consequences from $a < b$ and $(x < y) \wedge (y < z) \supset (x < z)$ despite the obvious fact that a refutation based on just these two formulas is impossible. A solution to this problem is to require that use of the transitivity axiom be restricted to occasions when either there are matches for two of its literals (partial theory resolution) or a complete refutation of the ordering part of the clauses can be found (total theory resolution).

An important form of reasoning in artificial intelligence applications embodied in knowledge representation systems [8] is reasoning about taxonomic information and property inheritance. One of our goals is to be able to take advantage of the efficient reasoning provided by a knowledge representation system by using it as a taxonomy decision procedure in a larger deduction system. This makes sense because it relieves the general-purpose deduction system of the need to perform taxonomic reasoning and because it extends the power of the knowledge representation system towards greater logical completeness. Other researchers have also cited the advantages of integrating knowledge representation systems with more general deduction systems [7, 26]. KRYPTON [6, 24] represents an approach to constructing a knowledge representation system composed of two parts: a terminological component (the TBox) and an assertional component (the ABox). For such systems, theory resolution indicates in general how information can be provided to the ABox by the TBox and how it can be used by the ABox.

2 Theory Resolution

We will now define the theory resolution operation and, in succeeding sections, discuss various useful restrictions on theory resolution. We will limit our discussion to the variable-free "ground" case of theory resolution, since lifting to the general case is straightforward. In the general case, it is required only that all ground theory resolvents of instances of the formulas be represented as instances of general theory resolvents.

We will assume the standard definitions of a *term*, an *atomic formula (atom)*, and a *literal*. In forming literals from atomic formulas, the symbol $\neg$ will be used to represent negation. It is also used to represent the operator that forms a literal of the opposite *polarity*, e.g., if L is the literal $\neg A$ then $\neg L$ denotes A.

We will consider a *clause* to be a disjunction of $n \geq 0$ literals. If $n = 0$, the clause is the *empty clause* $\square$. If $n = 1$, the clause is a *unit clause*. The disjunction connective $\vee$ is assumed to be associative and commutative, i.e., ordering of the literals in a disjunction is immaterial; $\vee$ is also assumed to be idempotent, i.e., forming the disjunction of two clauses which contain literals in common will result in only one occurrence of each such literal appearing in the resulting disjunction. The empty clause $\Box$ is the identity element for $\vee$. We will generally make no distinction between a unit clause and the single literal of which it is composed.

We will also use the standard $\wedge$ (conjunction), $\supset$ (implication), and $\equiv$ (equivalence) connectives in examples.

We will assume the standard definitions of an *interpretation*, an interpretation *satisfying* or *falsifying* a formula or set of formulas, and a formula or set of formulas being *satisfiable* or *unsatisfiable*.

Any satisfiable set of formulas that we wish to incorporate into the inference process can be regarded as a *theory*. This definition of a theory being a set of formulas is used in defining T-unsatisfiability, etc., but should not be taken too seriously for implementation purposes. An objective of theory resolution is the incorporation of theories into the inference process so that it will be unnecessary to resolve directly upon the axioms of the incorporated theory. Moreover, theory resolution does not require any direct representation of the formulas of the theory in the definition of the inference operations either. For example, the decision procedure that theory resolution requires can be a computer program, and the formulas that comprise the theory could only be ascertained by examining the structure and behavior of the program.

Definition 1 A *T-interpretation* is an interpretation that satisfies [the formulas of] theory T.

For example, in a theory of partial ordering ORD consisting of $\neg(x < x)$ and $(x < y) \wedge (y < z) \supset (x < z)$, the predicate $<$ cannot be interpreted so that $a < a$ has value $true$ or $a < c$ has value $false$ if $a < b$ and $b < c$ both have value $true$. In a taxonomic theory TAX including $Boy(x) \supset Person(x)$, $Boy(John)$ cannot have value $true$ while $Person(John)$ has value $false$.

Definition 2 A set of clauses S is *T-unsatisfiable* iff no T-interpretation satisfies S. S is *minimally T-unsatisfiable* iff S, but no proper subset of S, is T-unsatisfiable.

Definition 3 Let $C_1, \ldots, C_m$ $(m \geq 1)$ be a set of nonempty clauses, let each C_i be decomposed as $K_i \vee L_i$ where K_i is a nonempty clause, and let $R_1, \ldots, R_n$ $(n \geq 0)$ be unit clauses. Suppose the set of clauses $K_1, \ldots, K_m, R_1, \ldots, R_n$ is T-unsatisfiable. Then the clause $L_1 \vee \cdots \vee L_m \vee \neg R_1 \vee \cdots \vee \neg R_n$ is a *theory resolvent using theory T (T-resolvent)* of $C_1, \ldots, C_m$. The theory resolvent is called an *m-ary* theory resolvent (*unary* iff $m = 1$, *binary* iff $m = 2$). It is a *total* theory resolvent iff $n = 0$; otherwise it is *partial*. $K_1, \ldots, K_m$ is called the *key* of the theory resolution operation. For partial theory resolvents, $R_1, \ldots, R_n$ is a set of *conditions* for the T-unsatisfiability of the key. The negation $\neg R_1 \vee \cdots \vee \neg R_n$ of the conjunction of the conditions is called the *residue* of the theory resolution operation. It is a *narrow* theory resolvent iff each K_i is a unit clause; otherwise it is *wide*.

The definition above classifies individual theory resolution operations according to whether they are total or partial and wide or narrow. Theory resolution procedures are classified according to the operations allowed: partial theory resolution permits total as well as partial theory resolution operations, while total theory resolution permits only total theory resolution operations; wide theory resolution permits narrow as well as wide theory resolution operations, while narrow theory resolution permits only narrow theory resolution operations. Thus partial theory resolution includes total theory resolution and wide theory resolution includes narrow theory resolution.

Example 4 A set of unit clauses is unsatisfiable in the theory of partial ordering ORD iff it contains a chain of inequalities $t_1 < \cdots < t_n (n \geq 2)$ such that either t_1 is the same as t_n or $\neg(t_1 < t_n)$ is also one of the clauses. P is a unary total narrow ORD-resolvent of $(a < a) \vee P$. $P \vee Q$ is a binary total narrow ORD-resolvent of $(a < b) \vee P$ and $(b < a) \vee Q$. $P \vee Q \vee R \vee S$ is a 4-ary total narrow ORD-resolvent of $(a < b) \vee P$, $(b < c) \vee Q$, $(c < d) \vee R$, and $\neg(a < d) \vee S$. This can also be derived incrementally through partial narrow ORD-resolution, i.e., by resolving $(a < b) \vee P$ and $(b < c) \vee Q$ to obtain $(a < c) \vee P \vee Q$, resolving that with $(c < d) \vee R$ to obtain $(a < d) \vee P \vee Q \vee R$, and resolving that with $\neg(a < d) \vee S$ to obtain $P \vee Q \vee R \vee S$.

Example 5 Suppose the taxonomic theory TAX includes a definition for fatherhood $Father(x) \equiv [Man(x) \wedge \exists y Child(x, y)]$. Then $Father(Fred)$ is a partial wide theory resolvent of $Child(Fred, Pat) \vee Child(Fred, Sandy)$ and $Man(Fred)$. Also, $\Box$ is a total wide theory resolvent of $Child(Fred, Pat) \vee Child(Fred, Sandy)$, $Man(Fred)$, and $\neg Father(Fred)$.

Thus, the type of reasoning that is employable in the decision procedure can be quite different from and more effective in its domain than resolution.

Note that the definition of a theory resolvent includes ordinary resolvents since the unsatisfiability of pairs of complementary literals implies their T-unsatisfiability.

The following proves the soundness of theory resolution.

Theorem 6 *Let T be a theory, S a set of clauses, and C a T-resolvent of S. Then every T-interpretation I that satisfies S also satisfies C.*

Proof: Let I be any T-interpretation that satisfies S. Consider any decomposition of clauses in S into K_i and L_i leading to T-resolvent C being $L_1 \vee \cdots \vee L_m \vee \neg R_1 \vee \cdots \vee \neg R_n$. Suppose I satisfies some L_i. Then I also satisfies C. Alternatively, suppose I falsifies every L_i. Because I satisfies S, I must satisfy every K_i. But since $K_1, \ldots, K_m, R_1, \ldots, R_n$ is T-unsatisfiable, I must falsify at least one R_j. Thus, I must satisfy at least one $\neg R_j$ and consequently C. ∎

Definition 3 states what can be inferred by theory resolution and Theorem 6 confirms that the inferences are all valid. However, not all instances of theory resolution satisfying the definition need actually be inferred for completeness. This is important because the definition of theory resolution is too general to be usefully applied directly. For example, if $C_1, \ldots, C_m$ is any T-unsatisfiable set of clauses then $\Box$ is always derivable from $C_1, \ldots, C_m$ in a single total wide theory resolution step with each C_i decomposed into $K_i = C_i$ and $L_i = \Box$. This will be necessary for some sets of clauses, but to require it always would make the definition of theory resolution essentially useless.

In the following sections, we will explore some possible restrictions on the definition of theory resolution that make it practical to apply while preserving completeness. The first restriction we will consider is an instance of total wide theory resolution. In this form of total wide theory resolution, clauses are divided into two subclauses—one using predicates in the theory, the other not. If for some set of clauses the conjunction of the theory subclauses (the key) is T-unsatisfiable, then the disjunction of the nontheory subclauses can be derived by theory resolution. This is a powerful and general procedure, but has the disadvantage that deciding T-unsatisfiability of sets of clauses may be too difficult. Certainly it is infeasible for some specialized reasoning procedures, such as taxonomic reasoning in knowledge representation systems, that are incapable of dealing with disjunction.

More generally, it is often convenient to be able to use a simpler decision procedure than one capable of deciding T-unsatisfiability of sets of clauses. Narrow theory resolution is proposed for such cases. In narrow theory resolution, only T-unsatisfiability of sets of literals, not clauses, must be decided. Total and partial narrow theory resolution are both possible. In total narrow theory resolution, the resolved-upon literals (the key) must be T-unsatisfiable. In partial narrow theory resolution, the key must be T-unsatisfiable only under some conditions. The negated conditions are used as the residue in the formation of the resolvent. Partial narrow theory resolution generalizes results presented in [31] by allowing resolution with any number of key literals, i.e., by not being limited to binary theory resolution.

2.1 Total Wide Theory Resolution

Suppose we have a decision procedure for T that is capable of finding all minimally T-unsatisfiable subsets of any set of clauses containing only predicates in T. This procedure could then be applied to S with all literals having predicates not in T removed. For each minimally T-unsatisfiable subset discovered, a total wide theory resolvent, which contains no occurrences of predicates in T, can be derived.

This is a particular case of wide theory resolution in which the literals that are resolved upon by means of theory resolution are determined by their predicate symbols. Following is a completeness proof for this form of wide theory resolution.

Lemma 7 *Let S be an unsatisfiable set of clauses and P a set of predicates. Let S be decomposed into $S = S_P \cup S_{\overline{P}}$ so that every clause in S_P has an occurrence of a predicate in P and no clause in $S_{\overline{P}}$ has an occurrence of a predicate in P. Each clause C_i in S_P is of the form $K_i \vee L_i$, where all predicates in K_i are in P and all predicates in L_i are not in P. Each K_i is nonempty, but L_i might be empty. Let X be the set of all clauses such that $C_{i_1}, \ldots, C_{i_m}$ are clauses in S_P, $K_{i_1}, \ldots, K_{i_m}$ is a minimally unsatisfiable set of clauses, and the clause in X is of the form $L_{i_1} \vee \cdots \vee L_{i_m}$. Then $S_{\overline{P}} \cup X$ is unsatisfiable.*

Proof: Because S is unsatisfiable, and by virtue of the completeness of A-ordered resolution,[1] there exists an A-ordered resolution refutation of S with predicates in P preceding predicates not in P in the A-ordering. The refutation contains a possibly empty set X' of derived clauses that contain no predicates in P but whose parents contain predicates in P. By removing the derivations of clauses in X' from the refutation of S, we obtain a refutation of $S_{\overline{P}} \cup X'$. Thus $S_{\overline{P}} \cup X'$ is unsatisfiable. When we look at the A-ordered derivation, it is apparent that each clause in X' must be of the form $L_{i_1} \vee \cdots \vee L_{i_m}$ and that the corresponding set $K_{i_1}, \ldots, K_{i_m}$ must be unsatisfiable. If $K_{i_1}, \ldots, K_{i_m}$ is minimally unsatisfiable, then the clause in X' also belongs to X. If $K_{i_1}, \ldots, K_{i_m}$ is not minimally unsatisfiable, then the clause in X' is still subsumed by (possibly identical to) an element of X. Because X contains each element of X' or an element that subsumes it, the unsatisfiability of $S_{\overline{P}} \cup X$ follows from the unsatisfiability of $S_{\overline{P}} \cup X'$. ■

Theorem 8 *Let S be a T-unsatisfiable set of clauses. Then there is a refutation of S (derivation of $\Box$ from S) using total wide theory resolution with theory T.*

Proof: Consider representing the theory T as a set of clauses also called T. Because S is T-unsatisfiable, $S \cup T$ is unsatisfiable. Let P be the set of predicates occurring in T. By the lemma, $(S \cup T)_{\overline{P}} \cup X$, which equals $S_{\overline{P}} \cup X$ since $T_{\overline{P}} = \emptyset$, is unsatisfiable. Each clause in X is of form $L_{i_1} \vee \cdots \vee L_{i_m}$ and depends on the unsatisfiability of a set of clauses

[1] A-ordered resolution [9, 18] is a refinement of resolution that permits resolution only on the literals of each clause whose atoms appear earliest in a fixed ordering (the A-ordering).

$K_{i_1}, \ldots, K_{i_m}$. Some of these clauses K_{i_j} may be in T. Then the remaining clauses K_{i_k} are T-unsatisfiable; the clause in X is a total T-resolvent of the clauses C_{i_k} and is thus derivable by total wide theory resolution with theory T. Because $S_{\overline{P}} \cup X$ is unsatisfiable, it can be refuted by ordinary resolution, a special case of total wide theory resolution. Thus S has a refutation using total wide theory resolution with theory T. ∎

There are limitations to the use of total theory resolution. The requirement that the decision procedure for the theory be capable of determining unsatisfiability of any set of clauses using predicates in the theory is quite strict. Reasoning about sets of clauses is probably an excessive requirement for such purposes as using a knowledge representation system as a decision procedure for taxonomic information, since such systems are often weak in handling disjunction. This tends to limit total resolution's applicability to building in mathematical decision procedures that handle disjunction. For example, a decision procedure for Presburger arithmetic (integer addition and inequality) might be adapted to meet the requirements for total theory resolution.

Some care must be taken in deciding what theory T to incorporate. The theory must be capable of deciding sets of clauses that are constructed by using any predicates appearing in T. Thus, if we try to use total theory resolution to build in the equality relation with equality substitutivity (i.e., $x = y \supset (P(\cdots x \cdots) \supset P(\cdots y \cdots))$ for each predicate P), the decision procedure will have to decide all of S.

There may be a large number of T-unsatisfiable keys that do not result in useful T-resolvents. It would be a worthwhile refinement to monitor the finding of T-unsatisfiable sets of clauses to verify that the substitutions made do not preclude future use of the T-resolvent. This is like applying a purity check in A-ordered resolution.

2.2 Narrow Theory Resolution

Narrow theory resolution is a form of theory resolution that requires a less complex decision procedure than does wide theory resolution. Unlike the latter, which must consider T-unsatisfiability of sets of clauses, narrow theory resolution considers T-unsatisfiability of sets of literals. Thus, the requirement for the decision procedure for T to handle disjunction is eliminated.

Two principal variations of narrow theory resolution are considered here: total narrow theory resolution and partial narrow theory resolution.

In total narrow theory resolution, if $C_1, \ldots, C_m$ is a set of clauses, each C_i is decomposed as $K_i \vee L_i$, where K_i is a single literal, and $K_1, \ldots, K_m$ is minimally T-unsatisfiable, then $L_1 \vee \cdots \vee L_m$ is a T-resolvent of $C_1, \ldots, C_m$. The procedure remains sound if $K_1, \ldots, K_m$ is nonminimally T-unsatisfiable. However, completeness generally requires the absence of extraneous literals.

Partial narrow theory resolution is more general and includes total narrow theory resolution as a special case.

In partial narrow theory resolution, if $C_1, \ldots, C_m$ is a set of clauses, each C_i is decomposed as $K_i \vee L_i$ where K_i is a single literal, $R_1, \ldots, R_n$ is another set of literals, and $K_1, \ldots, K_m, R_1, \ldots, R_n$ is minimally T-unsatisfiable, then $L_1 \vee \cdots \vee L_m \vee \neg R_1 \vee \cdots \vee \neg R_n$ is a T-resolvent of $C_1, \ldots, C_m$. $K_1, \ldots, K_m$ is called the *key set of literals* or *key*. $R_1, \ldots, R_n$ are conditions for the T-unsatisfiability of $K_1, \ldots, K_m$, i.e., if $n > 0$, then $K_1, \ldots, K_m$ is not T-unsatisfiable, but $K_1, \ldots, K_m$ conjoined with $R_1, \ldots, R_n$ is. $\neg R_1, \ldots, \neg R_n$ is called the *residue set of literals* or *residue.* The procedure remains sound if $K_1, \ldots, K_m, R_1, \ldots, R_n$ is nonminimally T-unsatisfiable. However, completeness again generally requires the absence of extraneous literals.

Sometimes it is onerous to have to decide T-unsatisfiability of arbitrarily large sets of literals, as total narrow theory resolution requires. In these situations, partial narrow theory resolution can be used. The decision procedure for T must recognize the *potential* T-unsatisfiability of a set of literals, i.e., the set of literals is T-unsatisfiable *under some conditions.* The partial narrow theory resolvent will include (as the residue) the negation of the conditions for the T-unsatisfiability of the set of literals resolved on (the key) in the partial narrow theory resolution operation. The residue must be removed by later resolution operations for the resolvent to be used in a refutation. Partial narrow theory resolution thus finds T-unsatisfiable sets of literals incrementally by resolving on key sets of literals, adding residue sets of literals in the resolvent, and then resolving these away in further resolution operations.

Since partial narrow theory resolution will often examine sets of clauses that are smaller than those used for total narrow theory resolution, it may be less expensive to compute partial narrow theory resolvents than total narrow theory resolvents. Offsetting this advantage is the risk that more of the resolvents produced will be useless because their residues cannot be resolved away.

We do not want to require the derivation of all partial narrow theory resolvents permitted by the definition. Since partial narrow theory resolution includes total narrow theory resolution, this would require the derivation of all total narrow theory resolvents, thereby defeating the purpose of partial theory resolution.

It would also require the derivation of obviously unnecessary resolvents. For example, we could resolve $(a < b) \vee P$ and $(c < d) \vee R$, since, under some conditions such as $(b < c) \wedge (d < a)$, $a < b$ and $c < d$ are T-unsatisfiable. But it would be silly to draw such inferences. If we permit inferences from $a < b$ and $c < d$, which have no terms in common, the theory resolution procedure would not be very useful. If resolving $a < b$ and $c < d$ were to actually lead to a refutation—i.e., conditions for their T-unsatisfiability do hold—then some of these conditions, e.g., $(b < c) \wedge (d < a)$, must have arguments in common with $a < b$ and $c < d$. We should restrict partial theory resolution to cases in which the literals are suitably related.

To justify such pragmatically necessary restrictions on theory resolution, we offer the following criterion for the selection of key sets of literals that provides a sufficient condition for the completeness of partial narrow theory resolution.

In essence, the key selection criterion requires that every T-unsatisfiable set of literals have one or more subset key sets of literals that can be T-resolved. For example, in theory ORD, in refuting sets of positive inequality literals, we might select only pairs of literals matching $x < y$ and $y < z$ as key sets of literals. Thus, in refuting the set $\{a < b, b <$

$c, c < d, d < a\}$, we would be permitted, for example, to resolve upon $a < b$ and $b < c$, but not $a < b$ and $c < d$. Key sets of literals have one or more residues associated with them such that every minimally T-unsatisfiable set includes a key with a residue that can be refuted by resolving away the literals in the residue. With literals matching $x < y$ and $y < z$ selected, it is sufficient to derive T-resolvents with residue $x < z$. For example, $a < b$ and $b < c$ can be T-resolved with $a < c$ as the result. This can then be resolved with $c < d$ to derive $a < d$ that can be resolved with $d < a$ to derive $\Box$.

Key selection criterion.

- For any minimally T-unsatisfiable set of literals S, there is at least one key set of literals K such that $K \subseteq S$. K has at least two literals (one literal if S has only one literal).[2] Each K is recognizable by the decision procedure for T and will comprise the key for possible theory resolution operations, if clauses containing the key literals are present.

- For any such key set of literals K, there is at least one, possibly empty, residue set of literals R such that $K \cup \neg R$ is minimally T-unsatisfiable, where $\neg R$ denotes the set $\{\neg R_1, \ldots, \neg R_n\}$ when $R = \{R_1, \ldots, R_n\}$. Each $\neg R$ is a set of conditions for the T-unsatisfiability of key set K. Each R is computed from K by the decision procedure for T and is used as a residue for theory resolution operations that resolve on key K.

- It must be the case that, for some key set of literals K and associated residue set of literals R, $(S - K) \cup \{\vee R\}$ is minimally T-unsatisfiable, where $\vee R$ denotes the clause $R_1 \vee \cdots \vee R_n$ when $R = \{R_1, \ldots, R_n\}$. This ensures that key selection and residue computation will be sufficient for completeness—any T-unsatisfiable set of literals S has a T-resolvent using a key $K \subseteq S$ and residue R computed from K such that the T-resolvent is contradicted by the remaining literals $S - K$.

[2] The requirement that K have at least two literals unless S has only one literal conveniently eliminates the possible derivation of unary theory resolvents with nonempty residue. The use of such valid but unnecessary resolvents in a derivation can be supplanted by the use of higher-arity theory resolution operations.

In total narrow theory resolution, we uniformly take the key K to be the entire minimally T-unsatisfiable set of literals S. The residue R is always empty.

In partial narrow theory resolution, we will try to minimize the number of residue sets of literals. Thus, for $K = \{a < b, b < c\}$ we might have residues $R_1 = \{a < c\}$, $R_2 = \{\neg(c < x_1), \neg(x_1 < a)\}$, $R_3 = \{\neg(c < x_1), \neg(x_1 < x_2), \neg(x_2 < a)\}$, etc. However, only R_1 need be used, since, in the theory T, R_1 implies every other R_i. R_1 can be regarded as the strongest consequence of $a < b$ and $b < c$ in theory T.

There may still be more than one partial theory resolution operation with the same key but with different residues, because the key can be extended to a T-unsatisfiable set in more than one way. For example, given

$$\begin{gathered} triangle(a, b, c) \\ triangle(d, e, f) \\ length(a) = length(d) \\ length(b) = length(e) \\ length(c) = length(f) \end{gathered}$$

we could infer, using a theory of geometry, each of

$$\begin{gathered} angle(a, b) = angle(d, e) \\ angle(a, c) = angle(d, f) \\ angle(b, c) = angle(e, f) \end{gathered}$$

as residue.

The following theorem proves the completeness of narrow theory resolution with arbitrary selection of key sets of literals satisfying the key selection criterion. The completeness of total narrow theory resolution follows as an immediate corollary because total narrow theory resolution is just a special case of narrow theory resolution, where the key set of literals is selected to be the entire T-unsatisfiable set of literals and the residue is empty.

Theorem 9 *Let S be a T-unsatisfiable set of clauses. Then there is a refutation of S (derivation of $\Box$ from S) using partial narrow theory resolution with theory T for arbitrary selection of key sets satisfying the key selection criterion.*

Proof: If $\Box \in S$, then S is trivially refuted.

Otherwise we will prove the theorem by induction on complexity measure $c(S)$, where $c(S) = (|S|, k(S))$, where $|S|$ is the number of clauses in S and $k(S)$ is the *excess literal parameter* [2]. The excess literal parameter is defined to be the number of literals (i.e., literal occurrences) in S minus $|S|$. The ordering of $c(S)$ is defined by $c(S_1) \prec c(S_2)$ iff $|S_1| < |S_2|$, or $|S_1| = |S_2|$ and $k(S_1) < k(S_2)$.

Case $c(S) = (m, 0)$. Every clause must be a unit clause. Because S is T-unsatisfiable, it must include a minimally T-unsatisfiable subset S'.

> *Subcase* $|S'| \leq 2$. By the key selection criterion, S' must be selected as a key. The empty clause $\Box$ is derivable in a single unary or binary T-resolution step from S' and hence from S.
>
> *Subcase* $|S'| > 2$. By the key selection criterion, there exists a key $K \subseteq S'$ with $|K| \geq 2$ and (possibly empty) residue R such that $S'' = (S' - K) \cup \{\vee R\}$ is minimally T-unsatisfiable. $c(S'') \prec c(S') \preceq c(S)$. Thus, by the induction hypothesis, $\Box$ is derivable from S''. Since $\vee R$ is a T-resolvent of $K \subseteq S$, $\Box$ is derivable from S.

Case $c(S) = (m, n)$, $n > 0$. Select a nonunit clause $C \in S$. Decompose C into unit clause A and clause B, i.e., $C = A \vee B$. Because S is T-unsatisfiable, both $S_A = (S - \{C\}) \cup \{A\}$ and $S_B = (S - \{C\}) \cup \{B\}$ are T-unsatisfiable. Both $c(S_A) \prec c(S)$ and $c(S_B) \prec c(S)$. Thus, by the induction hypothesis, there must exist derivations of $\Box$ from each of S_A and S_B.

Imitate the derivation of $\Box$ from S_B, using C instead of B. The result will be a derivation of either $\Box$ or A from S. In the latter case, extend the derivation of A from S to a derivation of $\Box$ from S by appending the derivation of $\Box$ from S_A. ∎

Corollary 10 *Let S be a T-unsatisfiable set of clauses. Then there is a refutation of S (derivation of $\Box$ from S) using total narrow theory resolution with theory T.*

Although the theorem proves completeness of narrow theory resolution, its proof does not preclude the need for tautologies in a refutation. Indeed, it is the case that tautologies may have to be retained for a refutation to be found.

Example 11 Let T be the theory in which P, Q, and R are all equivalent. Let S be $\{P \vee Q \vee R, \neg P \vee \neg Q \vee \neg R\}$. There is a single-step wide T-resolution refutation of S. However, although there do exist refutations of S by narrow T-resolution,[3] all require retention of tautologies, since all narrow T-resolvents of $P \vee Q \vee R$ and $\neg P \vee \neg Q \vee \neg R$ are tautologies.

The theorem justifies the sufficiency of dealing with only single pairs of literals at a time, i.e., using binary partial narrow theory resolution. In this type of theory resolution, every theory resolvent has two parents (or maybe one, with no residue), provided key literals are selected in conformance with the key selection criterion.

This is often a very useful instance of theory resolution. It includes resolving on complementary pairs of literals (ordinary binary resolution) and on T-unsatisfiable pairs of literals. It also includes formation of partial theory resolvents for selected pairs of potentially T-unsatisfiable pairs of literals, including, for example, the derivation of the ORD-resolvent $a < c$ from $a < b$ and $b < c$. Despite the usefulness of binary partial theory resolution, there will be occasions when it is better to use nonbinary partial theory resolution to result in a smaller search space.

Finally, note that heuristic restrictions of theory resolution (such as discarding all tautologies, not recognizing all cases of T-unsatisfiability, or not computing all residues), though incomplete, may be very useful in practice.

2.3 Theory Matings

The *theory matings* procedure is another method of incorporating theories that is similar to the total narrow theory resolution method, in the sense of imposing the same requirements on the decision procedure for T, i.e., determining T-unsatisfiability of sets of literals, but

[3] For example, resolve on P in (1) $P \vee Q \vee R$ and $\neg P$ in (2) $\neg P \vee \neg Q \vee \neg R$ to obtain (3) $Q \vee R \vee \neg Q \vee \neg R$; then resolve on P in (1) and $\neg Q$ in (3) to obtain (4) $Q \vee R \vee \neg R$; finally resolve on P in (1) and $\neg R$ in (4) to obtain (5) $Q \vee R$. There is a similar derivation of (6) $\neg Q \vee \neg R$. Just as (5) and (6) were derived from (1) and (2), the contradictory unit clauses R and $\neg R$ can be derived from (5) and (6).

that does not depend on performing resolution inference operations. Hence it also overcomes the difficulty in total narrow theory resolution of retention of tautologies.

The theory matings procedure is an extension of Andrews's matings procedure [3] (see also Bibel's connection method [4]).

Definition 12 Let $C_1, \ldots, C_m$ $(m \geq 1)$ be a set of clauses. Then each set of literals $K_1, \ldots, K_m$ such that each K_i is a literal of C_i is a *path* through $C_1, \ldots, C_m$.

A path consists of one literal from each clause; it can also be regarded as one row of the dual, disjunctive normal form of the set of clauses.

The most important theorem for the application of the matings procedure is the following.

Theorem 13 *Let S be a set of clauses. Then S is unsatisfiable iff every path through S contains a complementary pair of literals.*

This can be easily extended to the theory matings procedure that builds in a theory T.

Theorem 14 *Let S be a set of clauses. Then S is T-unsatisfiable iff every path through S contains a T-unsatisfiable set of literals.*

Proof:

If part. Soundness. Assume that every path through S contains a T-unsatisfiable set of literals and is thus T-unsatisfiable. For S not to be T-unsatisfiable, there must be a T-interpretation I that satisfies each clause in S. To satisfy each clause in S, I must satisfy at least one literal of the clause. It must thus satisfy at least one path through S. But since every path is T-unsatisfiable, there is no such I. Hence S is T-unsatisfiable.

Only if part. Completeness. Assume that S is T-unsatisfiable. Suppose some path through S does not contain a T-unsatisfiable set of literals. Then that path has a T-interpretation I that satisfies it. But I would also satisfy S, since it satisfies a literal of each clause of S. This contradicts the assumption that S is T-unsatisfiable. Hence every path through S must contain a T-unsatisfiable set of literals. ∎

2.4 Metatheory Resolution

We regard the "black box" nature of the decision procedure being built in by means of theory resolution to be an important aspect of theory resolution because it allows flexibility in implementation of the decision procedure by not requiring it to employ clause representation, resolution, or other arbitrary restrictions.

However, one interesting possibility for implementation of decision procedures for theory resolution entails using, in the decision procedure, a resolution theorem prover whose domain of discourse is the language of the outer theorem prover.

We could axiomatize the T-unsatisfiability relation and use these axioms together with information on what literals are present to identify, by means of resolution, prospective key sets of literals to resolve upon. This reasoning system could itself depend on additional layers of theory resolution to establish its own inference operations.

This procedure of having successive layers of deduction systems determining unsatisfiability in successive theories in order to apply resolution operations could be called *metatheory resolution*.

3 Examples of Theory Resolution

Theory resolution is a procedure with substantial generality and power. Thus, it is not surprising that many specialized reasoning procedures can be viewed as instances of theory resolution, perhaps with additional constraints governing which theory resolvents can be inferred. We believe that the success of these specialized reasoning procedures helps to validate the concept of theory resolution.

First of all, we should note that there is a relationship between theory resolution and **hyperresolution**. Although further constraints (e.g., on the polarity of the literals) are often prescribed, the essence of hyperresolution is the derivation of $L_1 \vee \cdots \vee L_m \vee R$ from the *electron* clauses $K_i \vee L_i$, where K_i is a literal and L_i is a [possibly empty] clause and the *nucleus* clause $\neg K_1 \vee \cdots \vee \neg K_m \vee R$, where R is a [possibly empty] clause. This corresponds to a theory resolution operation using theory T, where $\neg K_1 \vee \cdots \vee \neg K_m \vee R$ is a consequence of T, $K_1, \ldots, K_m$ is the key set of literals, and R is the residue.

Theory resolution is also related to **procedural attachment** [23], whereby expressions are "evaluated" to produce new expressions. Ordinary procedural attachment can be regarded as unary theory resolution. Theory resolution in general can be considered as an extension of the notion of procedural attachment to sets of literals. Where ordinary procedural attachment permits the replacement of $2 < 3$ by *true*, theory resolution, in effect, can attach a procedure to the $<$ relation that permits derivation of $a < c$ from $a < b$ and $b < c$.

The **$ASK/$TELL mechanism** in ITP [19] is a procedural-attachment mechanism for incorporating specialized reasoning procedures in a resolution theorem-proving program. The $TELL predicate is used to make an assertion to another deductive system; $ASK is used to ask a query of the other deductive system. The $ASK/$TELL mechanism has been implemented for a PROLOG interpreter so that specialized reasoning tasks (such as reasoning about taxonomies and orderings) can be programmed in PROLOG and be invoked by $TELL and $ASK literals in the resolution theorem prover. $TELL and $ASK are also used to send messages to and ask queries of the user. Nonunary theory resolution operations, with or without a residue, can often be implemented effectively by using $ASK predicates in hyperresolution nuclei. The $ASK literals determine the validity of the operation and compute the residue; the ordinary literals comprise the negated key set of literals.

Two previous refinements of resolution that resemble partial theory resolution are Z-resolution and U-generalized resolution.

Dixon's **Z-resolution** [14] is essentially binary total narrow theory resolution with the restriction that T must consist of a finite deductively closed set of 2-clauses (clauses with length 2). This restriction does not permit inclusion of assertions like $\neg Q(x) \vee Q(f(x))$, $\neg(x < x)$, or $(x < y) \wedge (y < z) \supset (x < z)$, but does permit efficient computation of T-resolvents (even allowing the possibility of compiling T to LISP code and thence to machine code). Z-factoring and Z-subsumption operations are also defined.

Harrison and Rubin's **U-generalized resolution** [16] is essentially binary partial narrow theory resolution applied to sets of clauses that have a unit or input refutation. They apply it to building in the equality relation, developing a procedure similar to Morris's E-resolution [21]. The restriction to sets of clauses having unit or input refutations eliminates the need for factoring and simplifies the procedure, but otherwise limits its applicability. No effort was made in the definition of U-generalized resolution to limit the applicability of T-resolution to reasonable cases (e.g., formation of an ORD-resolvent of $a < b$ and $c < d$ is permitted by the definition).

The **linked inference principle** by Wos et al. [37] is related to theory resolution in concept and purpose. The linked inference principle is a somewhat more conservative extension of resolution than theory resolution, since it stipulates that the theory will be built in by means of clauses designated as linking clauses. Theory resolution, on the other hand, allows the theory to be incorporated as a "black box" that determines T-unsatisfiability questions in an unspecified manner. This facilitates the use of other systems, which do not rely upon resolution or clause representation, to build in theories. Nevertheless, many instances of theory resolution can be usefully implemented in the manner of the linked inference principle. Since the implementation proposal for the linked inference principle is more concrete, Wos et al. have expended comparatively more effort in determining how inference using the linked inference principle is to be controlled, including defining linked variants of resolution refinements such as unit-resulting resolution and hyperresolution.

Paramodulation [36] can be viewed as an instance of binary partial theory resolution where T is the theory of equality EQ. $P(b)$ is an EQ-resolvent (paramodulant) of $P(a)$ and $a = b$. Digricoli's **resolution by unification and equality** [13] can be viewed similarly. $a \neq b$ is an EQ-resolvent (RUE-resolvent) of $P(a)$ and $\neg P(b)$. Morris's **E-resolution** [21] can be viewed as an instance of total theory resolution. $\square$ is a total EQ-resolvent (E-resolvent) of $P(a)$, $\neg P(b)$, and $a = b$.

The difference between these equality reasoning procedures, from the standpoint of theory resolution, is the selection of the key set of literals for theory resolution operations.

The minimally EQ-unsatisfiable set of literals $P(a)$, $\neg P(b)$, and $a = b$ was used for all three examples. In the paramodulation example, $P(a)$ and $a = b$ were used as the key and $P(b)$ was derived as the residue. In the resolution by unification and equality example, $P(a)$ and $\neg P(b)$ were used as the key and $a \neq b$ was derived as the residue. In the E-resolution example, the key set consisted of all of $P(a)$, $\neg P(b)$, and $a = b$, and the residue was empty.

Where T consists of ordering axioms, including axioms that show how ordering is preserved (such as $(x < y) \supset (Pos(x) \supset Pos(y))$ and $(x < y) \supset (x + z < y + z)$), T-resolution operations include Manna and Waldinger's **relation replacement rule** (e.g., $Pos(b)$ can be inferred from $Pos(a)$ and $(a < b)$ and **relation matching rule** [20] (e.g., $\neg(a < b)$ can be inferred from $Pos(a)$ and $\neg Pos(b)$), which are extensions of paramodulation and resolution by unification and equality respectively. T-resolution with ordering axioms is also similar to Slagle and Norton's reasoning with **partial ordering inference rules** [29]. Bledsoe and Hines's **variable elimination and chaining** [5] is a refined method for reasoning about inequalities that can be viewed partly as theory resolution for inequality with added constraints on theory resolution operations. The ORD-resolvent $a < c$ of $a < b$ and $b < c$ is a variable-elimination-procedure chain resolvent only if b is a shielding term (nonground term headed by an uninterpreted function symbol). The variable-elimination rule allows inferring ORD-resolvent $(a < b) \vee C$ from clause $(a < x) \vee (x < b) \vee C$ only if x does not occur in a, b, or C. It more generally allows replacement of multiple literals $a_i < x$ and $x < b_j$ in a clause by literals $a_i < b_j$. This result is obtainable by theory resolution if we include the axiom $\neg(x < min(x, y))$ and a rule to transform $min(a_{i_1}, a_{i_2}) < b_j$ to $(a_{i_1} < b_j) \vee (a_{i_2} < b_j)$.

Allen's **interval-based temporal logic** [1] is another prospective system for explanation and implementation by theory resolution. Allen enumerates the 13 mutually exclusive relations that can hold between any pair of intervals: *before*, *after*, *during*, *contains*, *overlaps*, *overlapped-by*, *meets*, *met-by*, *starts*, *started-by*, *finishes*, *finished-by*, and *equal*. For each pair of temporal relations r_1 and r_2 and intervals A, B, and C such that $r_1(A, B)$ and $r_2(B, C)$, interval-based temporal logic defines what temporal relations are possible between

A and C. For example, if A *meets* B and B is *during* C, then either A *overlaps* C, A is *during* C, or A *starts* C. This corresponds to the theory resolution operation of resolving $during(A, B)$ and $overlaps(B, C)$ to obtain $during(A, C) \vee overlaps(A, C) \vee starts(A, C)$.

It is useful to create predicates that denote each of the 2^{13} combinations of possible temporal relations between a pair of intervals so that uncertainty about which relation holds will be represented by a predicate rather than a disjunction. Thus, $during(A, B)$ and $overlaps(B, C)$ would be resolved to obtain $[during|overlaps|starts](A, C)$. With this approach, it is possible to relate all the possible relations between A and C at once with other relations mentioning A or C by using narrow instead of wide theory resolution.

Another application of theory resolution is Konolige's **resolution for modal logic of belief** [17]. Among other things, he defines a resolution procedure that eliminates modal belief literals by recognizing their unsatisfiability in a subordinate deduction and then resolving on them in the manner of theory resolution. A simple propositional example of this is the resolution of $\Box p \vee A$, $\Box(p \supset q) \vee B$, and $\neg\Box q \vee C$ to obtain $A \vee B \vee C$, since the conjunction of $\Box p$, $\Box(p \supset q)$, and $\neg\Box q$ is unsatisfiable. For reasoning in his modal logic of belief, Konolige envisages using a system organized on principles similar to metatheory resolution.

We have already suggested the importance of theory resolution for **taxonomic reasoning**. This is being explored in the KRYPTON knowledge representation system [6, 24]. Figures 1 and 2 contain proofs with and without KRYPTON terminological reasoning.[4] The problem is to prove that, if Chris has no sons and no daughters, then Chris has no children. The terminological information used in this problem through theory resolution includes the statements that boys are persons whose sex is male; girls are persons whose sex is female; "no-sons" are persons all of whose children are girls; "no-daughters" are persons all of whose children are boys; every person has a sex; and males and females are disjoint.

From these facts and the assertions that Chris is a no-son and a no-daughter, a refutation

[4] Pigman [24] also presents a KRYPTON proof for this problem, but it differs from this one, since KRYPTON then differed in what terminological information was built in and how.

	1.	$Boy(x) \supset Person(x)$
	2.	$[Boy(x) \wedge Sex(x,y)] \supset Male(y)$
	3.	$Girl(x) \supset Person(x)$
	4.	$[Girl(x) \wedge Sex(x,y)] \supset Female(y)$
	5.	$[NoSon(x) \wedge Child(x,y)] \supset Girl(y)$
	6.	$[NoDaughter(x) \wedge Child(x,y)] \supset Boy(y)$
	7.	$Person(x) \supset Sex(x, sk1(x))$
	8.	$Male(x) \equiv \neg Female(x)$
hypothesis	9.	$NoSon(Chris)$
hypothesis	10.	$NoDaughter(Chris)$
negated conclusion	11.	$Child(Chris, sk2)$
resolve 11 and 5, simplify by 9	12.	$Girl(sk2)$
resolve 12 and 3	13.	$Person(sk2)$
resolve 7 and 13	14.	$Sex(sk2, sk1(sk2))$
resolve 11 and 6, simplify by 10	15.	$Boy(sk2)$
resolve 15 and 2	17.	$Sex(sk2,x) \supset Male(x)$
resolve 14 and 17	18.	$Male(sk1(sk2))$
resolve 8 and 18	19.	$\neg Female(sk1(sk2))$
resolve 12 and 4	20.	$Sex(sk2,x) \supset Female(x)$
resolve 14 and 20, simplify by 19	21.	$\Box$

Figure 1: Resolution Proof for *Childless* Problem

hypothesis	9.	$NoSon(Chris)$
hypothesis	10.	$NoDaughter(Chris)$
negated conclusion	11.	$Child(Chris, sk2)$
resolve 11 and 9	12.	$Girl(sk2)$
resolve 11 and 10, simplify by 12	13.	$\Box$

Figure 2: KRYPTON Proof for *Childless* Problem

is completed starting with the negation of the desired conclusion that Chris has no children. $sk1$ and $sk2$ are Skolem functions.

The following table demonstrates the improvement resulting from incorporating Formulas 1–8 into KRYPTON through theory resolution. The proof strategies used and meaning of the statistics are essentially the same as described in Section 4.

Built In Axioms	Inputted Formulas	Derived Formulas	Retained Formulas	Successful Unifications	Time (seconds)	Proof Length
none	11	10	20	37	1.1	9
1–8	3	2	5	4	0.4	2

Theory resolution for taxonomic reasoning also incorporates many elements of reasoning with a **many-sorted logic**. Two recent proposals for many-sorted logic in resolution are by Walther [32, 34] and Cohn [10, 11].

In Walther's ΣRP-calculus (many-sorted resolution and paramodulation), each function symbol and variable has a declared sort. The many-sorted unification algorithm imposes the restriction that a variable can be bound only to a term whose sort (the sort of the head function symbol for nonvariable terms) is the same as or a subsort of the variable's sort. When two variables are being unified and one variable's sort is a subsort of the other's, the first variable must be substituted for the second. When the sort hierarchy is not a tree and two variables with different sorts, neither a subsort of the other, are being unified, zero or more unifiers are produced. Each unifier results from substituting for both of the variables a new variable whose sort is one of the maximal common subsorts of the sorts of the variables.

An assertion like $[Bird(x) \wedge Plant(y)] \supset \textit{Likes-to-eat}(x, y)$ could be represented in the ΣRP-calculus by $\textit{Likes-to-eat}(x, y)$ in which x is a variable with sort $Bird$ and y is a variable with sort $Plant$. The characteristic predicates $Bird$ and $Plant$ are omitted and resolution is restricted so that, for example, $\textit{Likes-to-eat}(x, y)$ is resolvable with $\neg\textit{Likes-to-eat}(u, v)$, in which u and v are variables with sort $Animal$ and $Grain$ respectively, only because birds are animals and grains are plants. In this example, x and v would be substituted for u and y respectively.

This effect could be achieved in theory resolution using formulas like $Likes\text{-}to\text{-}eat(bird(x), plant(y))$ and $\neg Likes\text{-}to\text{-}eat(animal(u), grain(v))$ in which a term includes its sort. Theory resolution for a taxonomic theory could make these two ordinarily nonunifiable formulas unifiable by *generalizing* unification and provide the same result as many-sorted unification's *restricting* unification of the ordinarily always unifiable $Likes\text{-}to\text{-}eat(x, y)$ and $\neg Likes\text{-}to\text{-}eat(u, v)$.

If the sort hierarchy is a tree and each term specifies its own sort and supersorts as in $Likes\text{-}to\text{-}eat(animal(bird(x)), plant(y))$ and $\neg Likes\text{-}to\text{-}eat(animal(u), plant(grain(v)))$, then ordinary unification suffices to impose the sort constraints.

Many-sorted logic can be extremely effective (e.g., see Walther's solution of Schubert's Steamroller problem [33], a problem we also discuss in the next section). However, there are some important assumptions associated with its use. One is the assumption of the nonemptiness of the sorts used. $P(x)$ and $\neg P(x)$ are not contradictory if x's sort is empty.

More restrictive in practice is the assumption that terms can be assigned their sorts *a priori*. For example, suppose $Tweety$ is declared to have sort $Animal$ of which $Bird$ is a subsort. The absence of the characteristic predicate $Bird$ makes $Bird(Tweety)$ inexpressible. Even if the $Bird$ predicate is included, assuming or even proving the formula $Bird(Tweety)$ has no effect on $Tweety$'s declared sort which is used to restrict the unification algorithm. Thus, only those sorts for which it is unnecessary to assume or prove some term is a subsort of its declared sort should be in the many-sorted resolution sort hierarchy. The remaining sorts should be handled assertionally (theory resolution for taxonomic reasoning as discussed previously can help make this efficient).

Cohn's many-sorted logic is both richer and more complicated. Variables are unsorted and atomic formulas can be determined by the logic to have value $true$, $false$, $undetermined$, or $error$, depending on the sorts of the arguments. Thus, no sorted or unsorted assertion like "all birds like to eat all plants" is necessary—this information can be included in the sort declaration of the *Likes-to-eat* predicate so that $Likes\text{-}to\text{-}eat(s, t)$ will evaluate to $true$ if s and t have sort $Bird$ and $Plant$ respectively. Resolution with such assertions is supplanted by evaluation of the predicate. This may result in a further reduction in the sizes of the set of assertions and search space, as in Schubert's Steamroller problem [12]. Additional complexity when compared with Walther's many-sorted logic arises from the requirement that, when literals are removed from a clause by resolution or evaluation, the sorts of the variables must be interpreted consistently with the possibly polymorphic sort declarations of the predicates and across variable occurrences.

4 Experimental Results

Although the relationship of theory resolution to many other extensions of resolution (as discussed in the preceding section) and experience with numerous small examples support the practical value of theory resolution, we will not elaborate on these, but will rather bolster our claim with an examination of experimental results for "Schubert's Steamroller" challenge problem.

Schubert's Steamroller problem is

> Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain-eating animal.

In reporting a solution to this problem, Walther [33] states, "this problem became well known, since, in spite of its apparent simplicity, it turned out to be too hard for existing theorem provers because the search space is just too big."

An English-language solution to the problem is

> Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Snails are much smaller than birds and like to eat some plants. Because birds do not like to eat snails, it must be the case that birds like to eat all plants, including grains. Wolves do not like to eat grain. Therefore they must like to eat all animals much smaller than themselves that like to eat some plants. Because foxes are much smaller than wolves and wolves

> do not like to eat foxes, it must be the case that foxes do not like to eat plants. Therefore foxes like to eat birds since birds are grain-eating animals that are much smaller than foxes.

We will present results for two formulations of Schubert's Steamroller problem that differ only in the conclusion. The axioms are shown in Figure 3. The objective is to refute, from Formulas 1–25, either (a) Formula 26 (and thus prove $\exists x \exists y[Animal(x) \wedge Animal(y) \wedge \mathit{Likes\text{-}to\text{-}eat}(x, y) \wedge \exists z[Grain(z) \wedge \mathit{Likes\text{-}to\text{-}eat}(y, z)]]$) or (b) Formulas 27–28 (and thus prove $\exists x \exists y[Animal(x) \wedge Animal(y) \wedge \forall z[Grain(z) \supset [\mathit{Likes\text{-}to\text{-}eat}(x, y) \wedge \mathit{Likes\text{-}to\text{-}eat}(y, z)]]]$). Proof (b) corresponds to Walther's [33]. We used $a\text{-}wolf$, $a\text{-}fox$, etc., in Formulas 19–24 as the Skolem constants introduced by the assertions $\exists x.Wolf(x)$, $\exists x.Fox(x)$, etc. h, i, and j are Skolem functions.

We present statistics on several solutions of Schubert's Steamroller problem found by our theorem prover [30] with progressively more axioms built in by theory resolution.

The same strategy was used for all of the proofs. Clausal connection-graph resolution was the principal inference rule. Factoring was not employed. Pure, variant, and tautologous formulas were eliminated.

Single literal formulas were used for both forward and backward demodulation. In a technical sense, however, this is not demodulation—which is really a rule for simplification by equalities—but it performs a similar function. For example, if A is an atomic assertion, it could be used to simplify formulas containing atom A', an instance of A, by replacing A' by $true$ in the formula. Similarly, $\neg A$ could be used to replace instances A' of A by $false$. These could be construed as demodulation by the equivalences $A \equiv true$ and $A \equiv false$, respectively. Depending on the polarity of the atoms, the effect is essentially unit subsumption by the single literal formula or unit resolution with it.

Heuristic search, guided by a simple weighted function of the deduction level of the parents and the expected size of the resolvent, was used to decide which inference operation should be performed next. The set of support strategy (with only Formula 26 or 28 supported) and an ordering strategy that designated which atoms in a formula could be resolved upon were used to limit the number of alternative inference operations.

1. $Wolf(x) \supset Animal(x)$
2. $Fox(x) \supset Animal(x)$
3. $Bird(x) \supset Animal(x)$
4. $Caterpillar(x) \supset Animal(x)$
5. $Snail(x) \supset Animal(x)$
6. $Grain(x) \supset Plant(x)$
7. $[Snail(x) \wedge Bird(y)] \supset Much\text{-}smaller(x, y)$
8. $[Caterpillar(x) \wedge Bird(y)] \supset Much\text{-}smaller(x, y)$
9. $[Bird(x) \wedge Fox(y)] \supset Much\text{-}smaller(x, y)$
10. $[Fox(x) \wedge Wolf(y)] \supset Much\text{-}smaller(x, y)$
11. $[Wolf(x) \wedge Fox(y)] \supset \neg Likes\text{-}to\text{-}eat(x, y)$
12. $[Wolf(x) \wedge Grain(y)] \supset \neg Likes\text{-}to\text{-}eat(x, y)$
13. $[Bird(x) \wedge Snail(y)] \supset \neg Likes\text{-}to\text{-}eat(x, y)$
14. $[Bird(x) \wedge Caterpillar(y)] \supset Likes\text{-}to\text{-}eat(x, y)$
15. $Caterpillar(x) \supset Plant(h(x))$
16. $Caterpillar(x) \supset Likes\text{-}to\text{-}eat(h(x))$
17. $Snail(x) \supset Plant(i(x))$
18. $Snail(x) \supset Likes\text{-}to\text{-}eat(i(x))$
19. $Wolf(a\text{-}wolf)$
20. $Fox(a\text{-}fox)$
21. $Bird(a\text{-}bird)$
22. $Caterpillar(a\text{-}caterpillar)$
23. $Snail(a\text{-}snail)$
24. $Grain(a\text{-}grain)$
25. $Animal(x) \supset [[Plant(y) \supset Likes\text{-}to\text{-}eat(x, y)] \vee$
    $[[Animal(z) \wedge Much\text{-}smaller(z, x) \wedge Plant(w) \wedge Likes\text{-}to\text{-}eat(z, w)] \supset Likes\text{-}to\text{-}eat(x, z)]]$

    Negated conclusion for Proof (a):

26. $\neg Animal(x) \vee \neg Animal(y) \vee \neg Grain(z) \vee \neg Likes\text{-}to\text{-}eat(y, z) \vee \neg Likes\text{-}to\text{-}eat(x, y)$

    Negated conclusion for Proof (b):

27. $[Animal(x) \wedge Animal(y)] \supset Grain(j(x, y))$
28. $\neg Animal(x) \vee \neg Animal(y) \vee \neg Likes\text{-}to\text{-}eat(y, j(x, y)) \vee \neg Likes\text{-}to\text{-}eat(x, y)$

Figure 3: Steamroller Axioms

In using theory resolution, connection graph links were created from key sets of literals in the theory being incorporated. Formulas 1–6 and 14–18 were implemented by binary total narrow theory resolution links and Formulas 7–13 were implemented by 3-ary total narrow theory resolution links. For example, $Wolf(t)$ and $\neg Animal(t)$ could be linked, and $Bird(t_1)$, $Snail(t_2)$, and $Likes\text{-}to\text{-}eat(t_1, t_2)$ could all be linked. Theory resolution was also used in demodulation—e.g., $Wolf(t)$ could be used to demodulate $Animal(t)$ to $true$. This was accomplished by automatically adding extra demodulators such as $Animal(t) \equiv true$ for $Wolf(t)$.

Following are the statistics for the various solutions of Schubert's Steamroller problem. The first group of statistics is for Proof (a) refuting Formula 26; the second is for Proof (b) refuting Formulas 26–27. Included in the statistics are the number of formulas inputted to the theorem prover, the number of formulas derived in the course of searching for a proof, the number of inputted and derived formulas still present when a proof was found, the number of successful unification attempts during the search for a proof (including unification during link inheritance), the time required for the proof (on a Symbolics 3600 personal LISP machine), and the length of the proof in resolution steps.

Built In Axioms	Inputted Formulas	Derived Formulas	Retained Formulas	Successful Unifications	Time (seconds)	Proof Length
none	26	4,518	72	245,820	5,694	61
1–6	20	2,751	119	57,100	1,261	27
1–10	16	257	23	3,602	88	32
1–13	13	215	37	3,353	73	32
1–14	12	115	23	2,494	54	32
1–18	8	310	20	11,130	224	17
none	27	?	?	?	?	?
1–6	21	7,037	30	172,338	11,027	34
1–10	17	243	23	5,793	140	21
1–13	14	216	20	5,836	143	21
1–14	13	151	19	5,840	124	21
1–18	9	155	29	5,822	138	17

Thus, very substantial reductions in both the size of the search space and the length of the proof resulted from building in some of the axioms by theory resolution. This use of theory resolution has transformed a very difficult problem (for this deduction system) into a fairly easy one and must be considered highly successful. This is still not the optimal approach however. The use of many-sorted logic, whose assumptions match the problem requirements exceptionally well, would result in a solution with a search space measured in tens of unifications [33, 12]. This suggests that the particular form of taxonomic reasoning used here should be reserved for occasions when taxonomic reasoning by many-sorted logic (which can also often be regarded as an instance of theory resolution) is not applicable.

5 Conclusion

Theory resolution is a set of complete procedures for incorporating decision procedures into resolution theorem proving in first-order predicate calculus. Theory resolution can greatly decrease the length of proofs and the size of the search space. Total theory resolution can be used when there exists a decision procedure for the theory that is capable of determining unsatisfiability of any set of clauses using predicates of the theory. This may be a realistic requirement in some mathematical theorem proving.

Partial theory resolution requires much less of the decision procedure. It requires only that conditions for unsatisfiability of sets of literals be determinable by the decision procedure for the theory. This makes it feasible, for example, to consider use of a knowledge representation system as the decision procedure for taxonomic information.

Theory resolution is also a generalization of several other approaches to building in nonequational theories.

We are implementing and testing forms of theory resolution in the deduction-system component of the KLAUS natural-language-understanding system [15, 30]. This system demonstrated substantial improvement in performance when theory resolution was used on Schubert's Steamroller challenge problem. The KRYPTON knowledge representation system [6, 24] is also applying the ideas of theory resolution to combine a terminological reasoning component and an assertional reasoning component (for which they are also utilizing the KLAUS deduction system).

Theory resolution is a procedure with substantial power and generality. It is our hope that it will serve as a base for the theoretical and practical development of a methodology for combining the general reasoning capabilities of resolution theorem-proving programs with more efficient specialized reasoning procedures.

One important area for further research on theory resolution is finding restrictions on the need for retention of tautologies and determining compatibility with other resolution refinements.

Another important research question is handling combinations of theories (beyond the trivial case of totally disjoint theories). Successful combining of multiple deductive specialists within a resolution framework awaits further development in this area. The work of Nelson and Oppen [22] and Shostak [27] on combining quantifier free theories may be relevant.

Acknowledgments

The author would like to thank Mabry Tyson, Richard Waldinger, and Christoph Walther for their very useful comments on an earlier draft of this paper.

References

[1] Allen, J.F. Maintaining knowledge about temporal intervals. *Communications of the ACM 26,* 11 (November 1983), 832–843.

[2] Anderson, R. and W.W. Bledsoe. A linear format for resolution with merging and a new technique for establishing completeness. *Journal of the ACM 17,* 3 (July 1970), 525–534.

[3] Andrews, P.B. Theorem proving via general matings. *Journal of the ACM 28,* 2 (April 1981), 193–214.

[4] Bibel, W. *Automated Theorem Proving.* Friedr. Vieweg & Sohn, Braunschweig, West Germany, 1982.

[5] Bledsoe, W.W. and L.M. Hines. Variable elimination and chaining in a resolution-based prover for inequalities. *Proceedings of the 5th Conference on Automated Deduction,* Les Arcs, France, July 1980, 70–87.

[6] Brachman, R.J., R.E. Fikes, and H.J. Levesque. KRYPTON: a functional approach to knowledge representation. *IEEE Computer 16,* 10 (October 1983), 67–73.

[7] Brachman, R.J. and H.J. Levesque. Competence in knowledge representation. *Proceedings of the AAAI-82 National Conference on Artificial Intelligence,* Pittsburgh, Pennsylvania, August 1982, 189–192.

[8] Brachman, R.J. and B.C. Smith (eds). Special issue on knowledge representation. *SIGART Newsletter 70* (February 1980).

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

[10] Cohn, A.G. *Mechanizing a Particularly Expressive Many Sorted Logic.* Ph.D. dissertation, University of Essex, Essex, England, January 1983.

[11] Cohn, A.G. Improving the expressiveness of many sorted logic. *Proceedings of the AAAI-83 National Conference on Artificial Intelligence,* Washington, D.C., August 1983, 84–87.

[12] Cohn, A.G. A note concerning the axiomatization of Schubert's steamroller in many sorted logic. Department of Computer Science, University of Warwick, Coventry, U.K. September 1984.

[13] Digricoli, V.J. Resolution by unification and equality. *Proceedings of the Fourth Workshop on Automated Deduction,* Austin, Texas, February 1979, 43–52.

[14] Dixon, J.K. Z-resolution: theorem-proving with compiled axioms. *Journal of the ACM 20,* 1 (January 1973), 127–147.

[15] Haas, N. and G.G. Hendrix. An approach to acquiring and applying knowledge. *Proceedings of the AAAI-80 National Conference on Artificial Intelligence,* Stanford, California, August 1980, 235–239.

[16] Harrison, M.C. and N. Rubin. Another generalization of resolution. *Journal of the ACM 25,* 3 (July 1978), 341–351.

[17] Konolige, K. *A Deduction Model of Belief and Its Logics.* Ph.D. dissertation, Stanford University, Stanford, California, June 1984.

[18] Loveland, D.W. *Automated Theorem Proving: A Logical Basis.* North-Holland, Amsterdam, The Netherlands, 1978.

[19] Lusk, E.L. and Overbeek, R.A. A portable environment for research in automated reasoning. *Proceedings of the 7th Conference on Automated Deduction,* Napa, California, May 1984, 43–52.

[20] Manna, Z. and R. Waldinger. Special relations in automated deduction. To appear in *Journal of the ACM.*

[21] Morris, J.B. E-resolution: extension of resolution to include the equality relation. *Proceedings of the International Joint Conference on Artificial Intelligence,* Washington, D.C., May 1969, 287–294.

[22] Nelson, G. and D.C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems 1,* 2 (October 1979), 245–257.

[23] Nilsson, N.J. *Principles of Artificial Intelligence.* Tioga Publishing Co., Palo Alto, California, 1980.

[24] Pigman, V. The interaction between assertional and terminological knowledge in KRYPTON. *Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems,* Denver, Colorado, December 1984.

[25] Plotkin, G.D. Building-in equational theories. In Meltzer, B. and Michie, D. (eds.), *Machine Intelligence 7,* Halsted Press, 1972, pp. 73–90.

[26] Rich, C. Knowledge representation languages and predicate calculus: how to have your cake and eat it too. *Proceedings of the AAAI-82 National Conference on Artificial Intelligence,* Pittsburgh, Pennsylvania, August 1982, 193–196.

[27] Shostak, R.E. Deciding combinations of theories. *Journal of the ACM 31,* 1 (January 1984), 1–12.

[28] Siekmann, J.H. Universal unification. *Proceedings of the 7th International Conference on Automated Deduction,* Napa, California, May 1984, 1–42.

[29] Slagle, J.R. and L.M. Norton. Experiments with an automatic theorem-prover having partial ordering inference rules. *Communications of the ACM 16,* 11 (November 1973), 682–688.

[30] Stickel, M.E. A nonclausal connection-graph resolution theorem-proving program. *Proceedings of the AAAI-82 National Conference on Artificial Intelligence,* Pittsburgh, Pennsylvania, August 1982, 229–233.

[31] Stickel, M.E. Theory resolution: building in nonequational theories. *Proceedings of the AAAI-83 National Conference on Artificial Intelligence,* Washington, D.C., August 1983, 391–397.

[32] Walther, C. A many-sorted calculus based on resolution and paramodulation. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence,* Karlsruhe, West Germany, August 1983, 882–891.

[33] Walther, C. A mechanical solution of Schubert's steamroller by many-sorted resolution. *Proceedings of the AAAI-84 National Conference on Artificial Intelligence,* Austin, Texas, August 1984, 330–334.

[34] Walther, C. Unification in many-sorted theories. *Proceedings of the 6th European Conference on Artificial Intelligence,* Pisa, Italy, September 1984.

[35] Wos, L., R. Overbeek, E. Lusk, and J. Boyle. *Automated Reasoning.* Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[36] Wos, L. and G.A. Robinson. Paramodulation and set of support. *Proceedings of the Symposium on Automatic Demonstration,* Versailles, France, 1968. In *Lecture Notes in Mathematics 125,* Springer-Verlag, Berlin, West Germany, 1970, pp. 276–310.

[37] Wos, L., R. Veroff, B. Smith, and W. McCune. The linked inference principle, II: the user's viewpoint. *Proceedings of the 7th International Conference on Automated Deduction,* Napa, California, May 1984, 316–332.