4

Building-in Equational Theories

G. D. Plotkin
Department of Machine Intelligence
University of Edinburgh

INTRODUCTION

If let loose, resolution theorem-provers can waste time in many ways. They can continually rearrange the multiplication brackets of an associative multiplication operation or replace terms t by ones like $f(f(f(t, e), e), e)$ where f is a multiplication function and e is its identity. Generally they continually discover and misapply trivial lemmas. Global heuristics using term complexity do not help much and *ad hoc* devices seem suspicious.

On the other hand, one would like to evaluate terms when possible, for example we would want to replace $5+4$ by 9. More generally one would like to have liberty to simplify, to factorise and to rearrange terms. The obvious way to deal with an associative multiplication would be to imitate people, and just drop the multiplication brackets. However used or abused the basic facts involved in such manipulations form an equational theory, T, that is, a theory all of whose sentences are universal closures of equations.

Under certain conditions, we will be able to build the equational theory into the rules of inference. The resulting method will be resolution-like, the difference being that concepts are defined using provable equality between terms rather than literal identity. Therefore the set of clauses expressing the theory will not be among the input clauses, so no time will be wasted in the misapplication of trivial lemmas, since the rules will not waste time in this way.

Such devices as evaluation, factorisation and simplification consist of the application of a recursive function, N, from terms to terms with the property:

$$\vdash_T t = N(t).$$

For example, given an associative function f in t, N might rearrange t with all the f's as far to the right as possible; for an arithmetic expression t, $N(t)$ might be the value.

The simplification function can be extended to one from literals and clauses to, respectively, literals and clauses by:

$$N((\pm)P(t_1, \ldots, t_n)) = (\pm)P(N(t_1), \ldots, N(t_n))$$
$$N(C) = \{N(L) | L \in C\}$$

and similarly to other expressions.

We shall look for a complete set of rules $r_1, \ldots, r_m$ such that $r_1 \circ N, \ldots, r_m \circ N$ is also a complete set. (If r is a rule, $r \circ N$ is the rule which outputs $N(C)$ iff r outputs, with the same inputs, C). So the required facilities can be used, but incorporated in a theoretical framework.

While one could just drop the brackets of an associative operation, and make appropriate changes, such a procedure could not be systematic. Instead note that the set of terms in right associative normal form is in a natural 1–1 correspondence with the set of 'terms' in which the multiplication brackets and symbols, and the commas are dropped. So the bracketless terms will be considered simply as an abbreviation for the normal form. This situation seems to be quite common. Another example: if T is the theory of Boolean algebra, terms can be represented as sets of sets of literals, being in 1–1 correspondence, according to convention, with terms in either a suitable disjunctive or conjunctive normal form.

Normal forms will be considered to be given by a simplification function, N, which in addition satisfies the postulate:

If $\vdash_T t = u$ then $N(t)$ is identical to $N(u)$.

A theory T has a normal form iff it is decidable.

This does not cover all uses of the phrase. For example in the untyped λ-calculus, not all terms have one and N is only partial recursive. And of course things other than terms can have normal forms. As defined, normal form functions give less redundancy than any other simplification function.

The special characteristic of resolution-like methods is the unification algorithm. In our case, unification is defined relative to the theory T. A substitution, σ, T-unifies two terms t and u iff $\vdash_T t\sigma = u\sigma$. The exact generalisation of the notion of most general unifier will be given later; in general rather than a single most general unifier, there will be an infinite set of maximally general ones.

Define an N-application and composition operator, $*$, by:

$$t * \sigma = N(t\sigma)$$
$$\sigma * \mu = N(\sigma\mu)$$

It can be shown that $*$ possesses properties analogous to ordinary application and composition independently of the particular T and N chosen (see Robinson 1965).

For each equational theory one must invent a special unification algorithm with these equations built in. If we know a normalising function for the theory we are likely to get a more efficient algorithm. The purpose of this paper is not to display such algorithms, although we will give a couple of examples to fix ideas. Our purpose is to show how, given any such unification algorithm satisfying certain conditions, the usual resolution techniques can

be carried through using the more sophisticated algorithm, and without losing completeness.

Now as an example, consider the case where T is the theory of an associative function, f, whose language may include other function symbols, and $N(t)$ is the right associative form of t.

How can we unify $f(x, y)$ with $f(a, f(b, c))$ which is equal to $f(f(a, b), c)$? There are two T-unifications: $x=a$, $y=f(b, c)$ and $x=f(a, b)$, $y=c$. The first is straightforward. The second can be obtained in two steps:

(1) Put $x=f(a, x')$ where x' is a new variable.

(2) Unify $f(f(a, x'), y)$ with $f(a, f(b, c))$.

In step (2) we should normalise both expressions before unifying, so we actually unify $f(a, f(x', y))$ with $f(a, f(b, c))$. Thus $x'=b$ and $y=c$, giving us the second unification.

In practice it is simpler to reuse the variable x instead of introducing a new variable x'.

The unification algorithm is non-deterministic; a substitution is in the maximally general set of T-unifiers it defines iff the substitution can be output by the algorithm. That this set has the correct properties, and the correctness of the other algorithm given in this introduction is proved in Plotkin (1972). Terms are, temporarily, regarded as strings of function symbols, brackets, commas and variables. The disagreement pair of two terms t and u is the rightmost pair of distinct terms t' and u', such that t and u have the forms $Xt'Y$ and $Xu'Z$ respectively, for suitable strings X, Y and Z.

(1) Set σ equal to ε.

(2) If t is identical to u, stop with success and output σ.

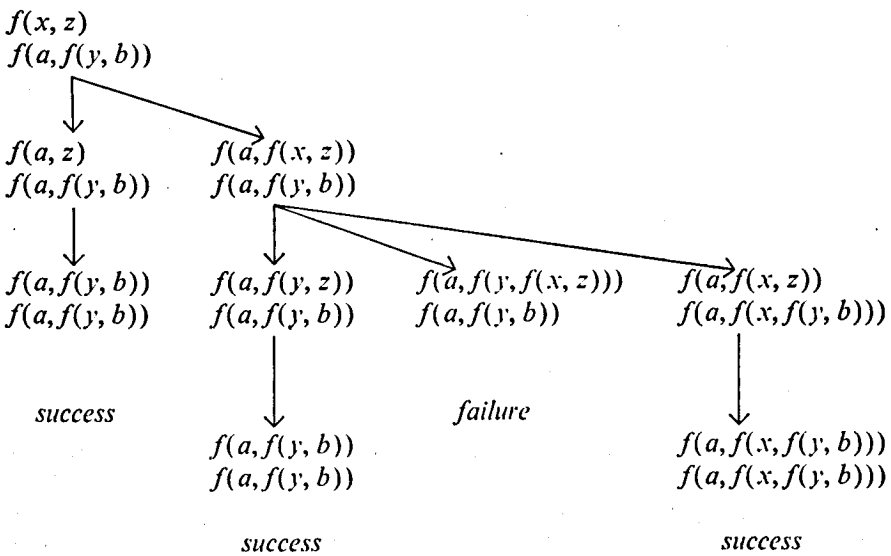

Figure 1

(3) Find the disagreement pair, t', u' of t and u.
(4) If t' and u' begin with different function symbols, stop with failure.
(5) If t' and u' are both variables, change t, u and σ to $t * \lambda$, $u * \lambda$ and $\sigma * \lambda$ where λ is either $\{u'/t'\}$, $\{f(u', t')/t'\}$ or $\{f(t', u')/u'\}$.
(6) If t' is a variable and u' is not, then, if t' occurs in u', stop with failure; otherwise change t, u and σ to $t * \lambda$, $u * \lambda$ and $\sigma * \lambda$ where λ is either $\{u'/t'\}$ or $\{f(u', t')/t'\}$.
(7) If u' is a variable and t' is not, then, if u' occurs in t', stop with failure; otherwise change t, u and σ to $t * \lambda$, $u * \lambda$ and $\sigma * \lambda$ where λ is either $\{t'/u'\}$, or $\{f(t', u')/u'\}$.
(8) Go to 2.

Figure 1 gives an example which traces the values of t and u through the execution tree of the algorithm when $t=f(x, z)$ and $u=f(a, f(y, b))$, giving all the successful and some of the unsuccessful searches.

The set of unifiers is $\{\{a/x, f(y, b)/z\}, \{f(a, y)/x, b/z\}, \{f(a, x)/x, f(x, y)/y, f(y, b)/z\}\}$.

One can have infinitely many successes, as illustrated in figure 2, where we are now using the abbreviation mechanism, writing xy for $f(x, y)$; g is some other function. In this case, the set of unifiers produced is

$$\{\{a^n/x, a^n/y\} \mid n>0\}.$$

Infinite failures are also possible: for example, take

$t=g(x, xa)$ and $u=g(y, by)$.

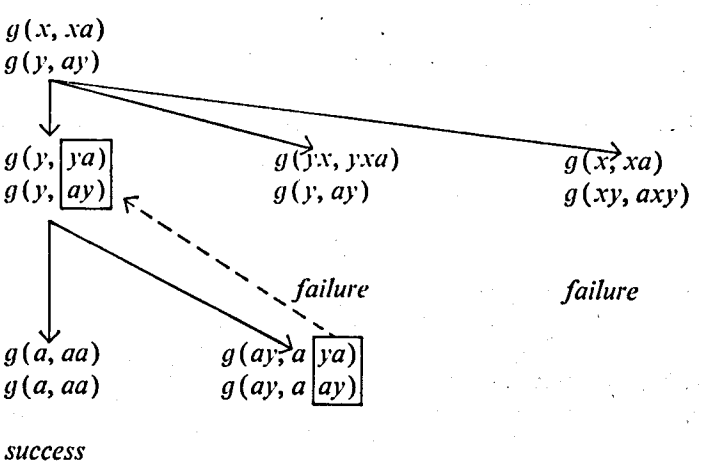

Figure 2

We conjecture that it is possible to decide whether two terms have a unifier – this is essentially the same problem as deciding whether a set of simultaneous equations in a free semigroup has a solution.

As a second example, we give an 'arithmetical evaluation' theory. Among its function symbols are two binary ones + and × and infinitely many

constants – the numerals $\Delta_n (n \geqslant 0)$. The axioms are:
(1) $+(\Delta_m, \Delta_n) = \Delta_{m+n} \qquad (m, n \geqslant 0)$
(2) $\times(\Delta_m, \Delta_n) = \Delta_{m \times n} \qquad (m, n \geqslant 0)$.
These are just the addition and multiplication tables.

$N(t)$ is obtained from t by continual application of the reduction rules:
(1) Replace an occurrence of $+(\Delta_m, \Delta_n)$ by one of $\Delta_{m+n} \qquad (m, n \geqslant 0)$
(2) Replace an occurrence of $\times(\Delta_m, \Delta_n)$ by one of $\Delta_{m \times n} \qquad (m, n \geqslant 0)$

A term, t, is a *number* term iff the only function symbols occurring in t are $+$, $\times$ or numerals.

Suppose Eq is a set of equations and Ineq is a set of sets of inequations and all terms in Eq or Ineq are number terms. A substitution, σ, *solves* Eq and Ineq iff:
(1) If x is a variable in Eq and Ineq then $x\sigma$ is a numeral and otherwise $x\sigma$ is x.
(2) If $t = u$ is in Eq then $t * \sigma$ is identical to $u * \sigma$.
(3) Every member of Ineq has a member, $t \neq u$, such that $t * \sigma$ is distinct from $u * \sigma$.

For example, $\{3/x\}$ solves $(\{x+2=5\}, \{\{x \neq 0\}\})$.

In general $\{\sigma \mid \sigma$ solves Eq and Ineq$\}$ is a partial recursive set, being the set of solutions of a certain set of Diophantine equations and inequations, effectively obtainable from Eq and Ineq.

The algorithm for producing a maximally general set of unifiers for two normal terms t and u is:
(1) Set Unify to $\{t = u\}$, Eq and Ineq to $\varnothing$ and σ to ε.
(2) Remove equations of the form $t = t$ from Unify.
(3) If Unify is empty, let μ be a substitution which solves Eq and Ineq, change σ to $\sigma * \mu$ and stop with success.
(4) Remove an equation, $t = u$, from Unify.
(5) If t and u are variables, replace Unify, Eq, Ineq and σ by Unify $* \lambda$, Eq $* \lambda$, Ineq $* \lambda$ and $\sigma * \lambda$ respectively, where $\lambda = \{t/u\}$.
(6) Suppose t is a variable and u is a number term. If t occurs in u, put $t = u$ in Eq; otherwise replace Unify, Eq, Ineq and σ by Unify $* \lambda$, Eq $* \lambda$, Ineq $* \lambda$ and $\sigma * \lambda$ respectively where $\lambda = \{u/t\}$.
(7) Suppose t is a variable and u is a term, but not a number one. If t occurs in u, stop with failure; otherwise replace Unify, Eq, Ineq and σ by Unify $* \lambda$, Eq $* \lambda$, Ineq $* \lambda$ and $\sigma * \lambda$ respectively, where $\lambda = \{u/t\}$.
(8) If u is a variable and t is not, proceed as in steps 6 and 7, but reverse the roles of t and u.
(9) If t and u begin with distinct function symbols then if either of them are not number terms stop with failure, otherwise put $t = u$ in Eq.
(10) If t and u begin with the same function symbol, f, say, and one of them is not a number term, then add

$$t_1 = u_1, \ldots, t_n = u_n \text{ to Unify where } t = f(t_1, \ldots, t_n) \text{ and } u = f(u_1, \ldots, u_n)$$

(11) If t and u have the forms $g(t_1, t_2)$ and $g(u_1, u_2)$ respectively and both

are number terms, where g is either $+$ or $\times$, then either add $t=u$ to Eq and $\{t_1 \neq u_1, t_2 \neq u_2\}$ to Ineq or else add $t_1=u_1$ and $t_2=u_2$ to Unify.
(12) Go to 2.

In the above, steps 3 and 11 are non-deterministic. The equation removed in step 4 is to be chosen in some fixed way. The algorithm always terminates, and can be implemented efficiently (apart from the difficulty of having to solve arbitrary Diophantine equations!). It uses no special properties of $+$ or $\times$ and so can be adjusted to suit any similar evaluation theory.

For example, suppose t is $\times(x, x)$ and u is $\times(\Delta_4, \times(y, y))$, then, at the beginning of the algorithm, we have Unify$=\{\times(x, x)=\times(\Delta_4, \times(y, y))\}$, Eq$=$Ineq$=\varnothing$ and $\sigma=\varepsilon$. Next, the execution sequence splits at step (11). Either Unify becomes $\varnothing$, Eq becomes $\{\times(x, x)=\times(\Delta_4, \times(y, y))\}$, Ineq becomes $\{\{x \neq \Delta_4, x \neq \times(y, y)\}\}$ and σ still has the value ε, or else Unify becomes $\{x=\Delta_4, x=\times(y, y)\}$, Eq and Ineq remain at $\varnothing$ and σ at ε. In the first case, the algorithm stops successfully at step (3), via a μ of the form $\{\Delta_{2n}/x, \Delta_n/y\}$, where $n \geqslant 0$ and $n \neq 2$, and then $\sigma=\mu$. In the second case, supposing $x=\Delta_4$ to be chosen at step (2), after step (6) we have Unify$=\{\Delta_4=\times(y, y)\}$, Eq$=$Ineq$=\varnothing$ and $\sigma=\{\Delta_4/x\}$. After steps (9) and (3), this execution path terminates with $\sigma=\{\Delta_4/x, \Delta_2/y\}$. If the commutativity and associativity of $\times$ were also built-in, one would expect here the single result, $x=\times(\Delta_2, y)$.

As a final example, the theory of an associative, commutative, idempotent binary function also has a unification algorithm – although we only know an extremely inefficient one. In this theory every pair of terms has a finite maximally general set of unifiers. This can be considered as building in sets, to some extent.

We believe most common algebraic systems admit unification algorithms. In particular we believe one can build in bags, lists and tuples in this way (Rulifson 1970). Group theory seems difficult. In general, the problem of finding a maximally general set of unifiers resembles, but is easier than the problem of solving in a closed form, equations in the corresponding free algebra. Other people have designed unification algorithms (Bennett, Easton, Guard and Settle 1967, Gould 1966, Nevins 1971, Pietrzykowski 1971, Pietrzykowski and Jensen 1972) but have not demonstrated that their methods satisfy the independence condition or, with the exception of Pietrzykowski *et al.*, the completeness one (see later). Cook (1965) seems to be the first person to notice the usefulness of normal forms in the theorem-proving context.

FORMAL PRELIMINARIES

The formalism is that of Robinson (1965) and Schoenfield (1967). The two are compatible with some minor adjustments. Clauses should be regarded as abbreviations for a corresponding sentence. Schoenfield omits brackets in his terms; these should be put back.

We use the equality symbol in infix mode and regard $t=u$ and $u=t$ as indistinguishable, thus building-in symmetry. Although the logic is not sorted, only minor adjustments are necessary to accommodate disjoint sorts.

The letters $t, u, \ldots$ range over terms. An occurrence of t in v is indicated by $v(t)$; $v(u/t)$ indicates the term obtained from $v(t)$ by replacing that distinguished occurrence of t in v by one of u.

The letters $L, M, \ldots$ range over literals. The general form of a literal is $(\pm)\, P(t_1, \ldots, t_n)$. $\bar{L}$ is like L, but has opposite sign.

The letters $C, D, \ldots$ range over clauses. By convention, $C \vee L$ represents the clause $C \cup \{L\}$ and implies L is not in C. Similarly $C \vee D$ represents $C \cup D$, and implies $C \cap D = \varnothing$ and $D \neq \varnothing$. Equations are unit clauses, $\{t=u\}$; $\{t \neq u\}$ is the general form of an inequation. The clause $\bar{C}$ is $\{\bar{L} \mid L \text{ in } C\}$.

The letter S varies over sets of clauses and Eq varies over sets of equations. The letter $\mathscr{E}$ stands for a set of sets, each member of which is either a finite set of terms or a finite set of literals. Greek letters $\sigma, \mu, \ldots$ range over substitutions: ε is the empty substitution. If $\xi = \{y_1/x_1, \ldots, y_n/x_n\}$ and the y_i are all distinct then ξ is *invertible* and ξ^{-1} is defined to be $\{x_1/y_1, \ldots, x_n/y_n\}$. The letter ξ is reserved for invertible substitutions. With each pair of clauses C and D, an invertible substitution $\xi_{C,D}$ is associated in some standard way, and is such that $C\xi_{C,D}$ and D have no common variables and if x does not occur in C, $x\xi_{C,D}$ is x.

The letter V ranges over finite sets of variables. Var(- - -) is the set of variables in the syntactic entity - - -, which can be a term, literal, clause or sets of sets of ... such; substitutions may be applied to such entities in the natural fashion. The *restriction* of σ to V, $\sigma \restriction V$ is the substitution μ such that if x is in V, $x\mu = x\sigma$ and otherwise $x\mu = x$.

From now on we discuss a fixed equational theory T with language L. In order to allow for Skolem function symbols we make:

Assumption 1. L contains infinitely many function symbols of each degree, none of which occur in T.

Suppose L has the form $(\pm)\, P(t_1, \ldots, t_n)$ and M has the form $(\pm)\, Q(u_1, \ldots, u_m)$. Then if P is not the equality symbol, $\vdash_T L \equiv M$ iff P and Q are identical, L and M have the same sign and, $\vdash_T t_i = u_i$, $(1 \leqslant i \leqslant n)$.

Simplification and normal-form functions have been defined in the introduction. They can be applied in the natural way to literals, clauses, and sets of sets of ... such.

GROUND LEVEL

We will formulate a set of rules, complete at ground level.

Define an equivalence relation, $\sim$, between literals by:

If the equality symbol does not occur in L, then $L \sim M$ iff $\vdash_T L \equiv M$. If it occurs with the same sign in both, L is $(\pm)\, t=u$ and M is $(\pm)\, v=w$, then $L \sim M$ iff $\vdash_T (t=v) \wedge (u=w)$ or $\vdash_T (t=w) \wedge (u=v)$. Otherwise, $L \nsim M$.

A set of literals is a T-*unit* iff $L \sim M$ for any L, M in the set.

If $C'' \cup \bar{D}''$ is a T-unit containing no occurrence of the equality symbol, then $C' \cup D'$ is a *ground* T-*resolvent* of $C = C' \vee C''$ and $D = D' \vee D''$.

If $C = C' \vee t_1 = u_1 \vee \ldots \vee t_n = u_n (n > 0)$, $D = D' \vee D''$, $(\pm) P(v_1, \ldots, v_{j_0}, \ldots, v_m)$ is in D'', $\{t_i = u_i | i = 1, n\}$ and D'' are T-units and $\vdash_T v_{j_0} = w(t_1)$ then

$$C' \cup D' \cup \{(\pm) P(v_0, \ldots, v_{j_0-1}, w(u_1/t_1), v_{j_0+1}, \ldots, v_m)\}$$

is a *ground* T-*paramodulant* of C and D.

If $C = C' \vee t_1 \neq u_1 \vee \ldots \vee t_n \neq u_n (n > 0)$ and $\{t_i \neq u_i | i = 1, n\}$ is a T-unit and $\vdash_T t_1 = u_1$, then C' is a *ground* T-*trivialisation* of C.

These rules are, in general, semi-effective; they are clearly consistent. If, in an application of one of the above, $\#(C'') = \#(D'') = n = 1$ ($\#$ is the cardinality function) the application has *degree* 1.

Given a set, Eq, of ground equations, write $t \approx_1 u$ iff there is a term $w(v)$ and another v' such that $\vdash_T t = w(v)$, $\{v = v'\}$ is in Eq and $u = w(v'/v)$. Note that if $t \approx_1 u$ and $\vdash_T u = v$ then for some w, $v \approx_1 w$ and $\vdash_T w = t$, that if $\vdash_T t = u$ and $u_1 \approx_1 w$ then $t \approx_1 w$, and if $t_j \approx_1 u$ then $f(t_1, \ldots, t_n) \approx_1 f(t_1, \ldots, t_{j-1}, u, t_{j+1}, \ldots, t_n)$.

Now define $\approx$ by: $t \approx u$ iff there are $t_i (n \geqslant 1$ and $1 \leqslant i \leqslant n)$ such that

$$t = t_1 \approx_1 \ldots \approx_1 t_n \text{ and } \vdash t_n = u.$$

Lemma 1. $t \approx u$ iff $\vdash_{T \cup Eq} t = u$, if t and u are ground terms.

Proof. Evidently $t \approx u$ implies $\vdash_{T \cup Eq} t = u$. Conversely, the properties of $\approx_1$ ensure that $\approx$ is a congruence relation. With the usual square bracket notation for congruence classes, define an interpretation, $\mathscr{A}$, whose domain is the set of congruence classes and whose operations are given unambiguously by:

$$\mathscr{A}(f)([t_1], \ldots, [t_n]) = [f(t_1, \ldots, t_n)].$$

Every equation in T $\cup$ Eq is true under this interpretation, so as $\vdash_{T \cup Eq} t = u$, then $[t] = \mathscr{A}(t) = \mathscr{A}(u) = [u]$; that is, $t \approx u$.

Theorem 1. Suppose S is a non-empty set of non-empty ground clauses such that $S \cup$ T has no model. Then there is a derivation of the null clause from S using the rules given above, in which all applications of the rules have degree 1.

Proof. The proof is by induction on the excess literal parameter

$$l(S) = \sum_{C \in S} (\#(C) - 1) \geqslant 0.$$

When $l(S) = 0$, S is a set of unit clauses. Let Eq $\subseteq S$ be the set of equations in S. If every set of the form Eq $\cup \{t \neq u\} \cup$ T ($\{t \neq u\}$ in S) or the form Eq $\cup \{\{L\}\{M\}\} \cup$ T ($\{L\}, \{M\}$ in S) is satisfiable, then it can be shown that $S \cup$ T is satisfiable. Consequently a set with one such form is unsatisfiable.

In the first case, $\vdash_{Eq \cup T} t = u$ and so, with $\approx$ as in lemma 1, $t \approx u$. So, by successive ground T-paramodulations from Eq into $t \neq u$, one can obtain a clause $t' \neq u$ such that $\vdash_T t' = u$. From this the null clause is obtained by a ground T-trivialisation of degree 1.

The second case is similar but uses a ground T-resolution.

If $l(S)>0$, there is a literal L in a non-unit clause C in S. Applying the induction hypothesis to $S'=(S\setminus\{C\})\cup(C\setminus\{L\})$, we obtain a derivation of the null clause, by the above rules, from S'. So either there is a derivation of the null clause from S by the rules or there is one of $\{L\}$ by them, from S. In the second case, we obtain, by induction, a refutation of $S''=(S\cup\{L\})\setminus\{C\}$ and the proof is concluded.

Notice that in the above if S contains no occurrences of the equality symbol, then a refutation of S can use only ground T-resolution; similar results hold for other grammatical possibilities.

A clause C, is a *ground N-T-resolvent of D and E* iff $C=N(C')$ for some ground T-resolvent of D and E. *Ground N-T-paramodulation* and *trivialisation* are similarly defined.

A partial order $\precsim$ is defined between clauses by:

$C\precsim D$ iff there is a function ϕ: $C\to D$ such that $L\sim\phi(L)$ for every L in C.

Note that $N(C)\precsim C$, for any clause C

One can check that if $C\precsim C'$, $D\precsim D'$ and E' is a ground T-resolvent of C' and D' then either $C\precsim E'$, $D\precsim E'$, or $E\precsim E'$ for some ground T-resolvent E of C and D; similar results hold for the other two rules.

From these remarks, it follows that if S has a refutation by the T rules, it has one by the N-T ones; further remarks analogous to those made immediately after theorem 1 also apply. (Actually only degree 1 applications of the rules are necessary).

Greater freedom in the application of N is permissible. One can allow N to depend on the derivation of the clause to which it is being applied; perhaps N could even depend on the state of execution of the algorithm searching for a refutation.

UNIFICATION

A substitution, σ, is a T-*unifier* of $\mathscr{E}$ iff when t, u are terms in the same set in $\mathscr{E}$, $\vdash_T t\sigma=u\sigma$ and if L, M are literals in the same set in $\mathscr{E}$, $L\sigma\sim M\sigma$.

An equivalence relation between substitutions is defined by:

$\sigma\sim\mu\ (V)$ iff, for all variables, x, in $V\vdash_T x\sigma=x\mu$.

Note that, if $\sigma\sim\mu(V)$ then $\sigma\nu\sim\mu\nu(V)$ and if $\sigma\sim\mu\,(\mathrm{Var}(V\nu))$ then $\nu\sigma\sim\nu\mu(V)$ for any ν.

A set, Γ, of substitutions is a *maximally general set of* T-*unifiers* (MGSU) of $\mathscr{E}$ *away from* V iff:

(1) (Correctness). Every σ in Γ T-unifies $\mathscr{E}$.

(2) (Completeness). If σ' T-unifies $\mathscr{E}$, then there is a σ in Γ and a λ such that $\sigma'\sim\sigma\lambda\,(\mathrm{Var}(\mathscr{E}))$.

(3) (Independence). If σ and σ' are distinct members of Γ, then for no λ does $\sigma'\sim\sigma\lambda\,(\mathrm{Var}(\mathscr{E}))$.

(4) For every σ in Γ and x not in $\mathrm{Var}(\mathscr{E})$, $x\sigma=x$.

(5) For every σ in Γ, $\mathrm{Var}(\mathscr{E}\sigma)\cap V=\varnothing$.

Conditions 4 and 5 are technical: from every set satisfying the first three conditions one can effectively and, indeed, efficiently construct one satisfying them all. Note that conditions 2 and 3 use relative equivalence rather than equality. These conditions can be satisfied in cases where the corresponding ones with equality cannot, and we know of no example of the opposite situation.

We also know of no example of a theory T and an $\mathscr{E}$ and V for which there is no such Γ, although we expect that one exists.

However we make:

Assumption 2. There is a partial recursive predicate $P(\mathscr{E}, \sigma, V)$ such that $\Gamma(\mathscr{E}, V) = \{\sigma | P(\mathscr{E}, \sigma, V)\}$ is a MGSU for $\mathscr{E}$ apart from V.

On this basis, semi-effective proof procedures can be developed. Of course in each case we should look for efficient algorithms for generating Γ.

We conjecture that if L_1 and L_2 are any two languages, not necessarily satisfying assumption 1 and L_1 is the language of T then if there is a (partial recursive, recursive, effectively obtainable, efficiently obtainable) MGSU for any $\mathscr{E}$, whose vocabulary is in L_1 and any V then there is a (partial recursive, recursive, effectively obtainable, efficiently obtainable) MGSU for any whose vocabulary is in $L_1 \cup L_2$. This would eliminate the necessity for assumption 1 and would as a special case, yield unification algorithms for evaluation systems like the arithmetical one in the introduction. Indeed if we consider theories T_1, T_2 whose languages are L_1, L_2 respectively, such that $L_1 \cap L_2 = \varnothing$, it may be the case that MGSU's for $T_1 \cup T_2$ always exist (and are partial recursive, etc.) if the same holds for T_1 and T_2 individually.

The algorithms given in the introduction do not quite conform to the specifications in that they do not give unifiers for an arbitrary $\mathscr{E}$ and, further, violate condition 5, in general. However, this is easily corrected. Interestingly, instead of condition 5, they satisfy:

(5′) For every σ in Γ, $\mathrm{Var}(\mathscr{E}\sigma) \subseteq \mathrm{Var}(\mathscr{E})$.

This condition allows the subsequent theory to proceed with some modifications, but can't be fulfilled for some T's for which 5 can.

MGSU's are essentially unique. If Γ, Γ' are MGSU's for $\mathscr{E}$ away from V and V', say, there is a bijection $\phi: \Gamma \to \Gamma'$ such that for all σ in Γ there are λ, λ' such that $\sigma \sim \phi(\sigma)\lambda(\mathrm{Var}(\mathscr{E}))$ and $\phi(\sigma) \sim \sigma\lambda'$ $(\mathrm{Var}(\mathscr{E}))$.

Again, if $\mathscr{E}$ and $\mathscr{E}'$ have the same set of T-unifiers and variables, Γ is a MGSU for $\mathscr{E}$ away from V iff it is for $\mathscr{E}'$. This holds, in particular, if $\mathscr{E}' = N(\mathscr{E})$.

A change of variables in $\mathscr{E}$ causes a corresponding one in Γ. If Γ is a MGSU for $\mathscr{E}$ away from V then if ξ maps distinct variables of $\mathscr{E}$ to distinct variables, $\{(\xi^{-1}\sigma) | \mathrm{Var}(\mathscr{E}\xi) | \sigma \in \Gamma\}$ is a MGSU for $\mathscr{E}\xi$ away from V.

Before defining suitable generalisations of resolution and so on, one should decide between formulations with or without factoring. Usually the difference is one, merely, of efficiency; the justification lies in a theorem of Hart (1965) (see also Kowalski (1970)), that if σ is a m.g.u. of $\mathscr{E}$ and σ' is of $\mathscr{E}'\sigma$ then $\sigma\sigma'$ is of $\mathscr{E} \cup \mathscr{E}'$. Unfortunately the generalisation of this theorem fails. We can prove:

Theorem 2 (*Weak Refinement Theorem*). Suppose Γ is a MGSU for $\mathscr{E}_1$ apart

from $\mathrm{Var}(\mathscr{E}_2) \cup V$ and for each σ in Γ, Γ_σ is a MGSU for $\mathscr{E}_2\sigma$, apart from $\mathrm{Var}(\mathscr{E}_1\sigma) \cup V$.

Then, $\Gamma' = \{\sigma\mu \restriction \mathrm{Var}(\mathscr{E}_1 \cup \mathscr{E}_2) \mid \sigma \in \Gamma, \mu \in \Gamma_\sigma\}$ satisfies conditions 1, 2, 4 and 5 for being an MGSU for $\mathscr{E}_1 \cup \mathscr{E}_2$ away from V.

Proof. Conditions 1, 4 and 5 may be straightforwardly verified. To establish condition 2, suppose θ T-unifies $\mathscr{E}_1 \cup \mathscr{E}_2$. Then by condition 2 on Γ, there is a σ in Γ and a λ such that:

$$\theta \sim \sigma\lambda(\mathrm{Var}(\mathscr{E}_1)).$$

It can be assumed, without loss of generality, that $\lambda = \lambda \restriction \mathrm{Var}(\mathscr{E}_1\sigma)$. Now $\mathrm{Var}(\mathscr{E}_2) \cap \mathrm{Var}(\mathscr{E}_1\sigma) = \varnothing$ by hypothesis and condition 5 on Γ. So $\lambda' = \lambda \cup (\theta \restriction \mathrm{Var}(\mathscr{E}_2))$ is a substitution. Now:

$$\theta \sim \sigma\lambda' (\mathrm{Var}(\mathscr{E}_1 \cup \mathscr{E}_2)).$$

For if x is in $\mathrm{Var}(\mathscr{E}_1)$ then $\mathrm{Var}(x\sigma) \cap \mathrm{Var}(\mathscr{E}_2) = \varnothing$, so $x\sigma\lambda' = x\sigma\lambda$ and $\vdash_T x\sigma\lambda = x\theta$. If x is in $\mathrm{Var}(\mathscr{E}_2)$ but not in $\mathrm{Var}(\mathscr{E}_1)$ then $x\sigma = x$, by condition 4 on Γ and $x\lambda = x$ by assumption. So $x\lambda' = x(\theta \restriction \mathrm{Var}(\mathscr{E}_2)) = x\theta$.

Therefore $\sigma\lambda$ T-unifies $\mathscr{E}_1 \cup \mathscr{E}_2$, λ T-unifies $\mathscr{E}_2\sigma$ and so by condition 2 on Γ_σ, there is a μ in Γ_σ and a δ such that:

$$\lambda' \sim \mu\delta \, (\mathrm{Var}(\mathscr{E}_2\sigma))$$

It can be assumed without loss of generality that $\delta \restriction \mathrm{Var}(\mathscr{E}_2\sigma\mu) = \delta$. Now, $\mathrm{Var}(\mathscr{E}_1\sigma) \cap \mathrm{Var}(\mathscr{E}_2\sigma\mu) = \varnothing$, by hypothesis and condition 5 on Γ_σ. So $\delta' = \delta \cup (\lambda' \restriction \mathrm{Var}(\mathscr{E}_1\sigma))$ is a substitution and one can show much as above that:

$$\lambda' \sim \mu\delta' \quad (\mathrm{Var}((\mathscr{E}_1 \cup \mathscr{E}_2)\sigma))$$

Then $\sigma\lambda' \sim \sigma\mu\delta \quad (\mathrm{Var}(\mathscr{E}_1 \cup \mathscr{E}_2))$ and:

$$\theta \sim \sigma\lambda' \sim \sigma(\mu\delta') \sim (\sigma\mu \restriction \mathrm{Var}(\mathscr{E}_1 \cup \mathscr{E}_2))\delta' \quad (\mathrm{Var}(\mathscr{E}_1 \cup \mathscr{E}_2)),$$

concluding the proof.

When T is the theory of an associative, commutative idempotent function, one can find examples where condition 3 fails. We will therefore formulate the rules without using factoring. (Factoring would still result in completeness, but would cause redundancy which although eliminable by a subsumption-type check would probably cause more trouble than it was worth.) Condition 3 does hold, however, in the associative and evaluation examples given in the introduction.

Gould (1966) has defined, in the context of ω-order logic, the concept of a general matching set.

In our terms, a literal, M, is a T-*unification* of L and L' iff for some σ, $L\sigma \sim M \sim L'\sigma$.

Then, Δ is a *general matching set of literals* for L and L' away from V iff:

(1) Every member of Δ is a T-unification of L and L'.

(2) If M' is a T-unification of L and L', then there is a λ and an M in Δ such that $M\lambda \sim M'$.

(3) If M, M' are distinct members of Δ then for every λ, $M\lambda \nsim M'$.

(4) If M is in Δ, $\mathrm{Var}(M) \cap V = \varnothing$.

In our opinion, this concept cannot, in general, be made the basis of a complete inference system; a counterexample will be given later.

GENERAL LEVEL

We begin with the rules.

Suppose σ is in $\Gamma(\{C''\xi_{C,D} \cup \bar{D}''\}, \mathrm{Var}(C\xi_{C,D} \cup D))$ where $C=C' \vee C''$, $D=D' \vee D''$ and the equality symbol has no occurrence in C''. Then,

$C'\xi_{C,D}\sigma \cup D'\sigma$ is a T-*resolvent* of C and D.

A variable-term pair is *special* iff it is of the form $\langle x, x\rangle$ or else $\langle x, f(x_1, \ldots, x_{i-1}, t, x_{i+1}, \ldots, x_n)\rangle (n>0)$ and $\langle x, t\rangle$ is special and the x_i are distinct and not in t.

Given an occurrence of a term u in another v there is a unique, to within alphabetic variance, special pair $\langle x, t\rangle$ and a substitution η such that:

(1) $v=t\{u/x\}\eta$,

(2) $\{u/x\}\eta=\eta\{u/x\}$,

(3) x has the same occurrence in t as u in v, and

(4) $\mathrm{Var}(t) \cap \mathrm{Var}(v)=\varnothing$.

We assume available a function, *Sp*, from finite sets of variables to special pairs such that:

(1) There is an alphabetic variant of every special pair in $Sp(V)$.

(2) No two distinct members of $Sp(V)$ are alphabetic variants.

(3) If $\langle x, t\rangle$ is in $Sp(V)$, $\mathrm{Var}(t) \cap V=\varnothing$.

Suppose $C=C' \vee t_1=u_n \vee \ldots \vee t_n=u_n (n>0)$, $D=D' \vee D''$, $(\pm)\ P(v_1, \ldots, v_{j_0}, \ldots, v_m)$ is in D'', $\langle x, t\rangle$ is in $Sp(\mathrm{Var}(C\xi_{C,D} \cup D))$ and that $\mathscr{E}=\{\{t_i\xi_{C,D} | i=1, n\}, \{u_i\xi_{C,D} | i=1, n\}, D'', \{v_{j_0}, t\{t_1\xi_{C,D}/x\}\}\}$.

Then, if σ is in $\Gamma(\mathscr{E}, \mathrm{Var}(C\xi_{C,D} \cup D))$,

$C'\xi_{C,D}\sigma \cup D'\sigma \cup \{(\pm)P(v_1\sigma, \ldots, v_{j_0-1}\sigma, t\{u_1\xi_{C,D}/x\}\sigma, v_{j_0+1}\sigma, \ldots, v_m\sigma)$ is an *assisted primary* T-*paramodulant from C into D*.

Suppose that $C=C' \vee t_1 \neq u_1 \ldots \vee t_n \neq u_n (n>0)$ and σ is in $\Gamma(\{\{t_i | i=1, n\} \cup \{v_i | i=1, n\}\}, \mathrm{Var}(C))$; then, $C'\sigma$ is a T-*trivialisation* of C.

These three rules are evidently consistent and semi-effective.

For completeness a lifting lemma is needed.

Lemma 2

(1) Suppose E' is a ground T-resolvent of ground clauses $C\mu$ and Dv. Then there is a T-resolvent, E of C and D and a λ such that $E\lambda \subseteq E'$.

(2) Suppose E' is a ground T-paramodulant from the ground clause $C\mu$ into the ground clause Dv. Then there is an assisted primary T-paramodulant, E, from C into D and a λ such that $E\lambda \subseteq E'$.

(3) Suppose E' is a ground T-trivialisation of the ground clause $C\mu$. Then there is a T-trivialisation, E, of C and a λ such that $E\lambda \subseteq E'$.

Proof. Only the proof of part 2, the hardest one, will be given.

Let $\sigma'=(\xi_{C,D}^{-1}\mu \restriction \mathrm{Var}(C\xi_{C,D})) \cup (v \restriction \mathrm{Var}(D))$.

Then $C\xi_{C,D}\sigma'=C\mu$ and $D\sigma'=Dv$. So, as E' is a ground T-paramodulant of $(C\xi_{C,D})\sigma'$ and $D\sigma'$ there are subsets C', D' and D'' of C and D, respectively,

and terms $t_i, u_i\ (i=1, n)$, $t, u, v'_{j_0}, v'(t)$ and a literal $(\pm)P(v'_1, \ldots, v'_{j_0}, \ldots, v'_m)$ in $D''\sigma'$ such that:

$C\xi_{C,D}\sigma' = C'\xi_{C,D}\sigma' \vee \{(t_i = u_i)\xi_{C,D}\sigma' \mid i=1, n\}$,
$D\sigma' = D'\sigma' \vee D''\sigma'$,
$t_1\xi_{C,D}\sigma' = t$
$u_1\xi_{C,D}\sigma' = u$
$\vdash_T v'_{j_0} = v'(t)$,
$E' = C'\xi_{C,D}\sigma' \vee D'\sigma' \vee (\pm)P(v'_1, \ldots, v'_{j_0-1}, v'(u/t), v'_{j_0+1}, \ldots, v'_m)$,
$\{(t_i = u_i)\xi_{C,D}\sigma' \mid i=1, n\}$ and $D''\sigma'$ are T-units,
$C = C' \vee t_1 = u_1 \vee \ldots \vee t_n = u_n$,
$D = D' \vee D''$,
$\vdash_T t_i\xi_{C,D}\sigma' = t \quad (i=1, n)$ and
$\vdash_T u_i\xi_{C,D}\sigma' = u \quad (i=1, n)$.

As t is a subterm of v', there is a unique $\langle x, w\rangle$ in $Sp(\mathrm{Var}((C\xi_{C,D} \cup D))$ and an η such that:

$v' = w\{t/x\}\eta$, $\{t/x\}\eta = \eta\{t/x\}$ and x has the same occurrence in w as the distinguished occurrence of t in v'.

All the above equations hold if σ' is replaced by $\sigma'' = \sigma'\eta$, as $C\mu$ and Dv are ground.

Now, let $(\pm)P(v_1, \ldots, v_m)$ be a literal in D'' such that $(\pm)P(v_1, \ldots, v_m)\sigma' = (\pm)P(v'_1, \ldots, v'_m)$. We have,

$v_{j_0}\sigma'' = v'_{j_0}$,
$\vdash_T v'_{j_0} = v'(t)$ and
$v'(t) = w\{t/x\}\eta = w\{t_1\xi_{C,D}\sigma'/x\}\eta$
$= w\{t_1\xi_{C,D}/x\}\sigma'\eta \qquad (\text{as } \sigma' \restriction \mathrm{Var}(w) = \varepsilon)$
$= w\{t_1\xi_{C,D}/x\}\sigma''$.

Therefore, $\vdash_T v_{j_0}\sigma'' = w\{t_1\xi_{C,D}/x\}\sigma''$, and we have proved that σ'' T-unifies $\mathscr{E} = \{\{t_i\xi_{C,D} \mid i=1, n\}, \{u_i\xi_{C,D} \mid i=1, n\}, D'', \{v_{j_0}, w\{t_1\xi_{C,D}/x\}\}\}$. So there is a σ in $\Gamma(\mathscr{E}, \mathrm{Var}((C\xi_{C,D} \cup D))$ and a λ' such that:

$\sigma'' \sim \sigma\lambda' \ (\mathrm{Var}(\mathscr{E}))$.

One then finds, as in the proof of theorem 2, a λ such that

$\sigma'' \sim \sigma\lambda \ (\mathrm{Var}((C\xi_{C,D} \cup D))$.

Now $E = C'\xi_{C,D}\sigma \cup D'\sigma \cup \{(\pm)P(v_1\sigma, \ldots, v_{j_0-1}\sigma,\ w\{u_1\xi_{C,D}/x\}\sigma, v_{j_0+1}\sigma, \ldots, v_m\sigma)\}$, is an assisted primary T-paramodulant from C into D.

Since $w\{u_1\xi_{C,D}/x\}\sigma'' = w\{u_1\xi_{C,D}/x\}\sigma'\eta$
$= w\{u_1\xi_{C,D}\sigma'/x\}\eta \qquad (\text{as } \sigma' \restriction \mathrm{Var}(w) = \varepsilon)$
$= w\{u/x\}\eta$
$= v'(u/t)$,

$E\lambda = C'\xi_{C,D}\sigma\lambda \cup D'\sigma\lambda \cup \{(\pm)P(v_1\sigma\lambda, \ldots, v_{j_0-1}\sigma\lambda, w\{u_1\xi_{C,D}/x\}\sigma\lambda, v_{j_0+1}\sigma\lambda, \ldots, v_m\sigma\lambda)\}$
$\subseteq C'\xi_{C,D}\sigma'' \cup D'\sigma'' \cup \{(\pm)P(v_1\sigma'', \ldots, v_{j_0-1}\sigma'', w\{u_1\xi_{C,D}/x\}\sigma'', v_{j_0+1}\sigma'', \ldots, v_m\sigma'')\}$
$= C'\xi_{C,D}\sigma'' \cup D'\sigma'' \cup \{(\pm)P(v'_1, \ldots, v'_{j_0-1}, v'(u/t), v'_{j_0+1}, \ldots, v'_m)\}$
$= E'$, concluding the proof.

Theorem 3. If S is a non-empty set of non-empty clauses such that $S \cup T$ has no model, there is a derivation of the null clause from S using the rules given above.

Proof. First it is shown by induction on derivation size (that is, the number of applications of rules) that if there is a derivation of a clause E' from a set of ground clauses of the form $\bigcup_i (S\sigma_i)$, using the ground rules, then there is a general derivation of a clause E from S and a λ such that $E\lambda \subseteq E'$.

When the derivation has size zero this is immediate.

Otherwise either there is a derivation of a ground clause C' from a set of clauses of the form $\bigcup_i (S\sigma_i)$ and another of a ground clause D' from a set of the form $\bigcup_j (S\sigma'_j)$, each derivation having strictly smaller size than the main derivation under consideration and E' is a ground T-resolvent of C' and D', or a corresponding statement involving one of the other two ground rules is true.

Let us suppose the first case holds. Then, by induction there are general derivations of clauses C and D from S and, substitutions μ and ν such that $C\mu \subseteq C'$ and $D\nu \subseteq D'$. By a remark made after theorem 1 either $C\mu \subseteq E'$, $D\nu \subseteq E'$ or there is a ground T-resolvent E'' of $C\mu$ and $D\nu$ such that $E'' \subseteq E'$. In the first two subcases, we are finished with this case. In the third, by lemma 2.1 there is a T-resolvent, E, of C and D and a λ such that $E\lambda \subseteq E''$, concluding this case.

A similar proof works in the other two cases.

Now, as $S \cup T$ has no model, there are, by Herbrand's Theorem, substitutions σ_i, such that $\bigcup_i (S\sigma_i)$ is ground and $\bigcup_i (S\sigma_i) \cup T$ has no model. Hence by theorem 1, there is a ground derivation of the null-clause from $\bigcup_i (S\sigma_i)$. By the above, there is a general derivation of a clause E from S and a λ such that $E\lambda \subseteq \varnothing$. Then $E = \varnothing$, concluding the proof.

Kowalski (1968) proved that, when $T = \varnothing$ one need only paramodulate into the functional reflexivity axioms, not from them, and all other paramodulations can be primary. Theorem 3 strengthens this result: special pairs take the place of the functional reflexivity axioms. One can also insist on $P1$ and A-ordering restrictions, analogous to Kowalski's, without losing completeness.

Other refinements are possible. For example one can define an E-T-resolution rule and a corresponding E-T-trivialisation one analogous to E-resolution (Morris, 1969). These two rules form a complete set. If one regards an E-T-resolution as a many-input rule, rather than a deduction using several T-paramodulations and a T-resolution (and similarly for E-T-trivialisation), and says that a clause has *support* if it is in the support set or is obtained by a rule application with a clause with support in its input set, the resulting set of support strategy is complete. (Anderson (1970) showed that a stronger one is incomplete). Presumably, although we have

not checked, one can obtain systems analogous to the other ones for equality.

As in the ground case, when there is no occurrence of the equality symbol in S, there is a refutation using only T-resolution, with similar things holding for the other grammatical possibilities. When there are no equality symbols occurring in S, refinements analogous to all the usual refinements of the resolution principle hold.

Again, using a simplification function N, one can define N-T-resolution and so on and obtain N-T-versions of all the above, by methods like those used in the ground case.

We do not know how to formulate the paramodulation conjecture, as the obvious way fails.

If $C=C' \vee C''$ and σ is in $\Gamma(\{C\}, \mathrm{Var}(C))$, then if L is in C'', $C'\sigma \cup \{L\sigma\}$ is a T-*factor* of C, with *distinguished literal* $L\sigma$.

If $C'=C'' \vee t=u$ is a T-factor of C with distinguished literal $t=u$, $D'=D'' \vee (\pm)P(v_1, \ldots, v_{j_0}, \ldots, v_m)$ is a T-factor of D with distinguished literal $(\pm)P(v_1, \ldots, v_m)$, $w(t')$ is a term such that $\vdash_T v_{j_0}=w(t')$ and σ is in $\Gamma(\{\{t, t'\}\}, \mathrm{Var}(C'\xi_{C',D'} \cup D'))$ then

$$C''\xi_{C',D'}\sigma \cup D''\sigma \cup \{(\pm)P(v_1, \ldots, w(u\xi_{C,D}/t'), \ldots, v_m)\sigma\}$$

is a T-*paramodulant from C into D*.

However, suppose T is the theory of an associative function, then if $S=\{f(bx, x)\neq f(xa, caa), bc=ca\}$, T is unsatisfiable but has no refutation by T-resolution, T-trivialisation and T-paramodulation.

A clause C, is a T-tautology iff $\vdash_T C$. We don't know if any deletion strategies work, even when $T=\varnothing$. If there are non-equality literals L, M in C such that $L\sim M$, or a literal $t=u$ in C such that $\vdash_T t=u$, then C is a *weak* T-*tautology*. The usual deletion strategies altered to take account of the extra rules work in this case.

A clause $C=\{L_i\}$ T-*subsumes* another $D=\{M_j\}$ iff there is a σ such that $\vdash_T (\bigvee_i L_i\sigma) \supset (\bigvee_j M_j)$. We do not know if any deletion strategies work, even when $T=\varnothing$. If $C\sigma \subseteq D$ for some σ then C *weakly* T-*subsumes* D. Appropriate alterations of the usual deletion strategies work in this case.

Here is an example showing why we do not consider that the concept of a general matching set of literals (or whatever) can be made the basis of a complete system of inference.

Consider the rule:

Let $C'=C'' \vee L$ and $D'=D'' \vee L'$ be T-factors of clauses C and D, with distinguished literals L and L' respectively. If M is in a general matching set of literals for $L\xi_{C',D'}$, and $\bar{L}'$ away from $\mathrm{Var}(C'\xi_{C',D'} \cup D')$, σ is such that $L\xi_{C',D'}\sigma \sim M \sim \bar{L}'\sigma$, and the equality symbol does not occur in L, then $C''\xi_{C',D'}\sigma \cup D''\sigma$ can be deduced from C and D by the rule.

But if T is the theory of an associative binary function, recursive effectively obtainable general matching sets for L and L' away from V exist, for all L, L'

and V, and furthermore the required σ's in the definition of the rule can be calculated from $L\xi_{C',D'}$, M and L'.

However, if $S=\{\bar{P}(ax, aa), \bar{Q}(aa, ax), P(yz, x) \vee Q(y, xw)\}$ then $S \cup T$ is unsatisfiable but S has no refutation by the above rule.

EFFICIENCY

The most obvious difficulty is that there can be infinitely many T-resolvents, etc., of two clauses. This does not cause any difficulty as far as the Hart-Nilsson *theory* is concerned (Kowalski 1972). It may be possible in some cases to represent the infinite set by using terms with parameters. These have already been used informally in the introduction.

Sometimes, as when T is the theory of an associative function, the unifiers are generated as successful nodes in a search tree. In this case the unification search tree can be incorporated in the general search space and one can save the unused parts.

However, we would like to advance the thesis that in general these large numbers of unifiers are present in the usual search spaces – we have just brought them to the surface. Indeed we believe that our method can have advantages over the standard resolution one precisely analogous to those obtained by the resolution method over the Davis-Putnam one. We will defend our method in the case where the theory is that of an associative function; the T-search space will be compared to that generated by resolution, paramodulation and trivialisation.

It can be shown that the associative unification of two literals, say, may be simulated by successive paramodulations of the axiom of associativity into these literals followed by an ordinary unification. The complexity of the simulation is about the same as that of the unification, using a symbol count. So the T-search space is included in the usual one.

The resolution method has smaller redundancy and avoids more irrelevancies than the Davis-Putnam method. For example, if L and M are unifiable then in general $\{\bar{L}, M\}$ has exactly one resolution refutation, but infinitely many Davis-Putnam ones. The crudest resolution search procedure produces no irrelevancies but the Davis-Putnam will, in general. If L, M are not unifiable, this will be detected by the resolution method, but not by the Davis-Putnam one. These phenomena are manifestations of the most general unifier method.

Similarly, if L and M have T-unifications, $\{\bar{L}, M\}$ can have many fewer T-resolution refutations than the comparison system. For example $\{\bar{P}(ax), P(yb)\}$ have two T-unifiers ($\{a/y, b/x\}$ and $\{ay/y, yb/x\}$ but there are infinitely many standard refutations of $\{\bar{P}(ax), P(yb)\}$, which essentially produce the T-unifiers, $\{ay_1 \ldots y_n/y, y_1 \ldots y_m b/x \mid m \geqslant 0\}$. Each of these unifiers can be produced in many different ways, involving arbitrarily long detours of bracket swapping.

As another example, consider $\{\bar{P}(x, y, xy), P(z, w, wz)\}$ which has an

infinite set of refutations involving the T-unifiers $\{\{w^m/z, w^m/x, w^n/y, w^n/w\}|$ *m and n positive integers with greatest common divisor unity*$\}$. The standard method essentially produces the denser set:

$$\{\{w^m/z, w^m/x, w^n/y, w^n/w\}|m, n>0\}.$$

Sometimes if L and M have no T-unifier, the T-unification algorithm will stop, when the standard procedure does not – for example, if $L=P(xa)$ and $M=\bar{P}(yb)$; but generally it will also generate useless structures (though not so many).

We believe that these informal remarks can be converted into a rigorous proof of increased efficiency.

On the other hand, the associative unification procedure can certainly be greatly improved, and we have no practical experience at the moment. It is surely not the case that these methods will by themselves make a practical theorem-prover, of course. We have only removed one of many exponential explosions.

PROBLEMS

There are many obvious problems concerning particular axiom systems and how to combine different unification procedures.

However, what does one do with only partial knowledge? Suppose one has a simplification method, but no unification algorithm as is the case, at the moment, with group theory or with integration algorithms (Moses 1967)? Or, what use can be made of a one-way unification procedure? How and when does one simplify (Moses 1971)? Answering such questions might produce more efficient systems closer to normal human practice.

Acknowledgements

I have had many helpful discussions with Rod Burstall. Jerry Schwarz found some mistakes in a draft. The work was supported by the Science Research Council.

REFERENCES

Anderson, R. (1970) Completeness results for E-resolution. *Proc. AFIPS 1970 Spring Joint Comp. Conf.* Washington, DC.

Bennett, J.H., Easton, W.B., Guard, J.R. and Settle, L.G. (1967) C.R.T.-aided semi-automated mathematics. Final report. AFCRL-67-0167. Princeton: Applied Logic Corporation.

Cook, S.A. (1965) Algebraic techniques and the mechanisation of number theory. RM-4319-PR. California, Santa Monica: RAND Corporation.

Gould, W.E. (1966) A matching procedure for ω-order logic. *Sci. Rep. No. 4.* AFCRL 66-781. Princeton, New Jersey: Applied Logic Corporation.

Hart, T.P. (1965) A useful property of Robinson's unification algorithm. *A.I. Memo 91.* Project MAC. Cambridge, Mass.: MIT.

Kowalski, R.A. (1968) The case for using equality axioms in automatic demonstration. *Lecture Notes in Mathematics*, vol. 125. (eds Laudet, M., Nolin, L. and Schützenberger, M.) Berlin: Springer-Verlag.

Kowalski, R. (1970) Studies in the completeness and efficiency of theorem-proving by resolution. Ph.D. Thesis. Department of Computational Logic, University of Edinburgh.

Kowalski, R.K. (1972) And-or graphs, theorem-proving graphs and bi-directional search. Machine Intelligence 7, paper 10 (eds Meltzer, B. & Michie, D.). Edinburgh: Edinburgh University Press.

Morris, J.B. (1969) E-resolution: extension of resolution to include the equality relation. *Proc. First Int. Joint Conf. on Art. Int.* Washington, DC.

Moses, J. (1967) Symbolic integration. *Report* MAC-TR-47. Project MAC. Cambridge, Mass.: MIT.

Moses, J. (1971) Algebraic simplification: a guide for the perplexed. *Proc. Second Symp. on Symbolic and Algebraic Manipulation*, pp. 282–303. (ed. Petrich, S.R.). New York: Association for Computation Machinery.

Nevins, A.J. (1971) A human-oriented logic for automatic theorem proving. TM-62789. George Washington University.

Pietrzykowski, T. (1971) A complete mechanisation of second-order logic. Science Research Report (CSRR 2038). Department of Analysis and Computer Science, University of Waterloo.

Pietrzykowski, T. & Jensen, D.C. (1972) A complete mechanisation of ω-order logic. Science Research Report CSRR 2060. Department of Analysis and Computer Science, University of Waterloo.

Plotkin, G.D. (1972) Some unification algorithms. Research Memorandum (forthcoming). School of Artificial Intelligence, University of Edinburgh.

Robinson, J.A. (1965) A machine-oriented logic based on the resolution principle. *J. Ass. comput. Mach.*, **12**, 23–41.

Rulifson, J.F. (1970) Preliminary specification of the QA4 language. Artificial Intelligence Group. Technical Note 50. Stanford Research Institute.

Schoenfield, J.R. (1967) *Mathematical logic.* Reading, Mass.: Addison-Wesley.