A SIMPLIFIED FORMAT FOR THE MODEL ELIMINATION THEOREM-PROVING PROCEDURE

D. W. Loveland

Report 67-39

December, 1967

Abstract

This paper presents an alternate approach to the formulation of the Model Elimination proof procedure presented in [Ll] . By exploiting fully the ability to linearize the procedure format (isolating the format from a tree structure form) and by representing lemmas by clauses, the description of the Model Elimination procedure is greatly simplified.

A SIMPLIFIED FORMAT FOR THE MODEL ELIMINATION THEOREM-PROVING PROCEDURE

D. W. Loveland

§1. In [Ll] a proof procedure for first order predicate calculus is presented and arguments showing the possibility of increased efficiency over some existing procedures are offered. The format of the procedure in [Ll] is somewhat cumbersome, a fault which this paper attempts to overcome. In this paper, several related proof procedures are given of which the final procedure is equivalent to the *Model Elimination* procedure presented in [Ll]. This paper can be read independently of [Ll] although here only the description and proof of its soundness and completeness are given. [Ll] should be consulted for a non-trivial example of an appliction of the procedure as well as a discussion of the relative efficiency of the procedure. Knowledge of one of [Ll], [Rl] or [R2] is desirable for §1 and §2; 43 utilizes results from [Rl] and [R2] which are quoted without proof here. §3 contains the proofs of the lemma and theorems of ^2.

It is well known that the question of validity of a first order well-formed formula is equivalent to the corresponding question for the closure of the formula. Validity of a (closed) formula is equivalent to unsatisfiability of the formula^{1}s negation. We are concerned with establishing the unsatisfiability of a given closed formula. With no loss of generality, we may presume the formula to be in *Skolem functional form* (for satisfiability) ; that is, a (closed) well-formed formula (wff) in prenex normal form having only universal quantifiers and allowing constants and function symbols. We also include the requirement that the quantifier free *matrix* be in conjunctive normal form. The matrix then is a conjunction of *clauses* with each clause a disjunction of *literals*. A *clause instance* of a given clause C is a clause C6 where 0 is a *substitution* of terms (elements of a given set H) for some (perhaps all) of the variables of C. The same replacement must be used at each occurrence of a designated variable. The null substitution *e* such that Ce = C is permitted.

The procedures considered in this paper are built upon Herbrand's theorem. Herbrand* s theorem formulated for unsatisfiable sentences states that a closed wff W in Skolem functional form is unsatisfiable if and only if there are a finite number of clause instances of W whose conjunction is truth functionally unsatisfiable. The set H(W) of terms from which the clause instances of the theorem are derived is defined recursively as follows:

a) all variables of the logic and all constants of W are in H(W);

b) if $t_1,...,t_n$ are elements (*terms*) of H(W) and f is an n-place function symbol appearing in W, then f(t_x,•..,t_n) is in H(W);

c) only terms derivable from a) and b) appear in H(W).

A proof of this theorem appears in [R2]. The concern of the procedures presented here as with the other proof procedures based on Herbrand^{1}s theorem is the efficient and correct determination of the finite contradictory conjunction of clause instances whenever such a conjunction exists. As is frequently done, we identify a set of clauses with the conjunction of the members of the set. Thus we use phrases such as 'the set of matrix clauses' etc.

Recent Herbrand-type proof procedures including the procedures developed here employ a technique first applied to mechanical theorem proving by Prawitz. Robinson [Rl] employs the same principle; we follow his development. Let A.$_1$ and A$_0$ be two atomic formulas (*attorns*) . The set (A$_1$,A$_9$} is said to te *unifiable* if there exists a substitution 9 such that A$_1$.0 = A^G. The substitution 0 is then said to be the *unifier* of A,$_1$ and A~$_2$. The null substitution is permitted. We present an algorithm,, called the *unification algorithm* by Robinson, which by a series of substitutions in A,$_1$ and A~$_2$ unifies {A^A^} if unification is possible.

Unification Algorithm

1) Set two pointers, one on the first (leftmost) symbol of each atom A.$_1$ and A~$_2$.

2) Advance the pointers through each atom in parallel one symbol at a time until the first (next) *point of disa-*

greement is met, if any. If the pointers simultaneously reach the end of each atom without (further) disagreement, the algorithm terminates in mode S (success). Otherwise, go to 3) .

3) Unless at the point of disagreement one pointer points to a variable V and the other pointer points to the first symbol of a well formed expression (term) not containing V, terminate the algorithm in mode F (failure). Otherwise replace the variable V at each occurrence of V in both A_1 and A_2 by the term whose first symbol is at the point of disagreement. If both atoms have a variable at the point of disagreement let V be the variable of atom A_1. Return to 2).

Note that at each stage in the algorithm there is agreement in A_1 and A_2 to the left of the parallel pointers. Thus the algorithm has transformed A_1 and A_2 into identical atoms if the algorithm terminates in mode S. Using the fact that the composition of two substitutions is also a substitution,, it is seen that the Unification Algorithm, when successful,, defines a substitution a which we designate the mqsjt general unifier of {A_1.,A_2}. The justification for the name given a comes from the following theorem proved in [Rl].

Unification theorem (Robinson). If A_n and A~ are atoms and {A^A^} is unifiable, then there exists a most general unifier a of {$A_{13}A_9$}. Moreover, for any 0 that unifies {A_1,A_0} there exists a substitution A such that A^G = (A^cr) A -⁻ (A$_2$cx) X .

Two literals are said to be complementary if one is the negation of the other. For two given literals $L_1 J L_2$ if there exists a most general unifier *a* such that L_1cr and L_2a are complementary, then we say a match is possible for L_1 and L_2.

For any expression E and substitution 9 we call EG an instance or refinement of E. If V.. , . . .V, are the variables appearing in *E,* then by the x-instance of E we mean the expression $E£_E$ where the substitution $£_E$ replaces V_1 ...V_k by variables x_1,...x_k respectively. Likewise-, by the y~instance of E we mean the expression $Er]_E$ where $r)_E$ replaces V, $,_1$. . .V_k by variables y_1j,...j,y_k respectively.

Example of (most general) unification: For A_1 = P(x$_5$f(y, a)$_9$z)$_9$ A_2 = P(z,w,x) we have A_1a = *A*<$_2$ = P(z, f(y, a)$_9$z) whereas A_1 and AI_2 = Ptz^x^y) have no unifier.

§2. We define in this section three closely related procedures which deal with structures called chains. A chain, denoted by K or K_n for non-negative integer *n*$_9$ is a finite ordered list of literals. Two types of literals occur in a chain, class A literals (A-literals) and class B literals (B-literals). A non-negative integer called the deficit of K is associated with each chain K and used to indicate if a chain has been derived from a chain with more A-literals. An elementary chain is an ordered clause with all literals considered B-literals and a deficit of zero. Not all chains are equally desirable; we distinguish two subclasses. A chain is _preadmissible_ if the following three conditons are met:

a) if two B-literals are complementary they must be separated by an A-literal;

b) if a B-literal is identical to an A-literal the B-literal must precede the A-literal in the chain;

c) no two A-literals have identical atoms.

A chain is _admissible_ if it is preadmissible and the last element in the chain is a B-literal. The _empty chain_ 0, a chain containing no literals, is defined to be admissible.

New chains are produced from existing chains by use of three operations defined below. Each operation has an input chain called the _parent chain_; the Extension operation also has a second chain as input, always an elementary chain. The output chain is called the _derived chjain_ of the operation (or, alternately, _derived chain_ from the parent chain), The operations have the form of applying a substition 8 (possibly null) to the parent chain K then adding or deleting literals to obtain the derived chain. If L is a literal of K and L0 appears in the derived chain, then L is called the _parent literal_ and L9 its _derived literal_. A finite sequence $K_o, \ldots, K_n$ of chains such that K_{i+1} is a derived chain from K_i, $0 \leq i \leq n - 1$, is termed a _deduction_ of K_n from K_o. (The accompanying set of elementary chains is not acknowledged by this notation but will always be clear by context.) Within a given deduction, K_a

is an ancestor chain of K_{fe} if K^ precedes K^ in the deduction, K_{xD} is then a descendant chain from K_a . L_1 in K_a lb an ancestor literal of L_2 in K^ if there is a sequence of parent literal--derived literal relationships connecting L^ and L_2. L_9 is then the descendent literal of L_1. Note that if L_2 is a descendent of L_1 there exists a substitution A such that L_2 = L_1X but the converse need not be true.

We define the operations Basic Extension, Basic Reduction and Basic Contraction which are the operations used in the first two procedures to be defined. The prefix 'basic[1] distinguishes these operations from their slightly modified successors which incorporate a useful lemma device.

1) Basic Exte^ns^iqri. This operation has as input two chains, an admissible chain K as the parent chain and also an elementary chain C taken from the auxiliary set G of elementary chains. The x-instance K£__ of K and the y-instance $Cr\backslash_c$ of C are formed. If a match is possible between the last literal L_{JL}, of $K£_K$__ and the first literal L_2 of Crj_c, let a denote the associated most general unifier of the match. If a match is impossible the operation terminates. If a match is possible,, the derived chain is formed as follows: L_2cr (the first literal) is removed from the chain $C77^c$ cr and the remaining part of the chain is appended to $K£T_K$cr following the last literal L_1a

of *K£*$_K$a$_5$ preserving the order of *C77*$_C$a. The literal L_1cr becomes an A-literal. All other literals with parents in K receive their parent's classification. All appended literals derived from C are B-literals. If the deficit of K is positive, the deficit of the derived chain is one less than the deficit of K, otherwise the deficit is zero.

2) Basic Reduction. This operation has an admissible chain K as input. A match of literals {L_1,L_2} is sought for some A-literal L.$_1$ and a B-literal *"L*$_2$ following L.$_1$ in the order of K. If a match is impossible$_9$ the operation terminates. If a match is possible, then let a be the most general unifier such that L_1a and L~$_2$cr are complements. The derived chain is *YSJ* with L_2Cr deleted. The deficit of the derived chain is that of the parent chain. All*literals have the classification of their parent literals.

3) Basic Contraction. This operation has a preadmissible chain K as input. The derived chain is the parent chain with all A-literals that follow the last B-literal deleted. The deficit of the derived chain is that of K plus the number of A-literals deleted. All derived literals have the classification of their parent literals.

Note that except for the Basic Contraction operation, the derived chains may be non-preadmissible. The non-preadmissible chains may be discarded immediately as they are input chains to no operation. The Basic Contraction operation produces admissible chains given preadmissible chains; this operation is regarded as always performed immediately upon the creation of a preadmissible chain.

A *matrix chain* is an elementary chain formed from a matrix clause. For a matrix clause of n literals there are ni possible matrix chains. However, except for possible value for added heuristics, the distinction among matrix chains created from the same clause and having the same first element is unnecessary. Thus we let $\mathfrak{M}_o(S)$ represent a set of matrix chains formed from the given set S of matrix clauses such that there is precisely one chain in $\mathfrak{M}_o(S)$ from a given clause with a given first literal.

We present a procedure by defining sets C_n of chains such that $C_n \subseteq C_{n+1}$, $n = 0,1,2,3,\ldots$. The set C_n is regarded as the set of chains that have been produced by the conclusion of an *nth* stage of the procedure. The order of production of chains within a stage is of relative unimportance. Two of the procedures are now defined. These first two procedures are partial constructs of the third procedure which is the procedure of major interest. We also make use of the first procedure in the proof of completeness.

Definition. A deduction of a chain K from a chain C is *n-bounded* if no more than n (Basic) Extension operations are employed in the deduction. A deduction K is conditionally n-bounded if no more than n (Basic) Extension operations are employed on chains of zero deficit in the deduction.

First procedure:

$$C_0^{(1)} = \omega_0(S),$$

$C_n^{(1)} = \{K \mid K$ is an admissible chain and there is an n-bounded deduction of K from a chain of $\omega_0(S)$. The auxiliary set is $\omega_0(S)$.$\}$

Second procedure:

$$C_0^{(2)} = \omega_0(S),$$

$C_n^{(2)} = \{K \mid K$ is an admissible chain and there is a conditionally n-bounded deduction of K from a chain of $\omega_0(S)$. The auxiliary set is $\omega_0(S)\}$

It is easily shown that each set $C_n^{(2)}$ is finite.

Theorem 1. A set S of clauses is unsatisfiable if and only if there exists an N such that $C_N^{(i)}$ contains the empty chain, i = 1,2.

This theorem stated for the sequence $\{C_n^{(1)}\}$ is sufficient to establish the corresponding theorem for the sequence $\{C_n^{(2)}\}$ as for each n there is an m such that $C_n^{(2)} \subseteq C_m^{(1)}$.

The second procedure, however, states an important strategy employed in the Model Elimination procedure, namely, to give preference to deductions which give some indication that progress towards achieving the empty chain has been made (this corresponds to the 'partially contradictory set^1 notion of [Ll]). In the second procedure we treat as a special case each deduction whose A-literal count on the last chain of the deduction is strictly less than some earlier chain in the deduction. The procedure allows continued development within stage n of the special deductions as long as contractions occur sufficiently often to keep the A-literal count condition valid.

To define the third procedure the operations are altered to allow the introduction of lemmas. A *lemma* is an elementary chain. Lemmas, as formed, are 'added^1 to the set to$_o$(S) of matrix chains to form new auxiliary sets to be used with the Extension operation. A lemma can sometimes be used to extend a chain resulting (perhaps after use of the Reduction operation) in a preadmissible chain where no matrix chain would have led to a preadmissible chain at the same point. (See [Ll] for an example of the advantageous use of lemmas).

To aid in the production of lemmas, a non-negative integer is associated with each A-literal and called the *scope* of the A-literal. Every derived A-literal has its parent^{1}s scope unless otherwise stated. We introduce a set variable to(S) which shall denote the auxiliary set used in a particular application of the Extension operation. to(S) is regarded as containing the matrix chains and the lemmas already generated at the time the particular application of the Extension operation occurs. Because we have not specified a completely sequential development of chains (and the corresponding deductions) we approximate here this status by formally defining (a slightly more restricted) to(S) as follows (recall the definition is dependent on the point of application of the Extension operation): to(S) contains 1) the matrix chains of to_o(§), 2) the lemma chains created in the deduction $ of K where K is the chain to which the Extension operator is applied, 3) the lemma chains defined in $C^{(3)}_{n-1}$ if $ is a deduction of maximal length in $C^{(3)}_{n-1}$ or if $K \notin C^{(3)}_{n-1}$.

We define now the full operations of Extension, Reduction and Contraction.

1) Extension--the same as the basic extension operation except the auxiliary set is to(S) instead of to_o(S)[1]. The newly formed A-literal is assigned scope 0.

2) Reduction--the same as basic reduction except the scope of the A-literal responsible for the deletion of the B-literal may be increased. Let m be the scope of the parent A-literal involved. If n is the number of A-literals (strictly) between this A-literal and the B-literal to be removed and n > m, the scope of the derived A-literal is n.

3) Contraction--the basic Contraction operation is modified as follows. The A-literals to be removed are removed individually in reverse of chain order. After each remwal a chain (lemma) is formed consisting of the complement of the removed A-literal L_1 as the first element of the chain plus the complement of each preceding A-literai

L_o satisfying the following: the number of A-literals (strictly) between L_2 and L_1 is less than the scope of Lp. As with matrix literals, only the placing of the first element is significant, ordering the remaining A-literals (if any) in the lemma may be done by any convenient algorithm. Finally, every A-literal in the derived chain whose scope would otherwise exceed the number n of A-literals greater than that A-literal has its scope reduced to n.

Definition: By a full conditionally n-bounded deduction we mean a conditionally n-bounded deduction employing the full operations just defined in place of the basic operations,

Third procedure:

$$C_o^{(3)} = m_o(S)$$

$C_n^{(3)} = \{K \mid K$ is an admissible chain and there is a full conditonally n-bounded deduction of K from a chain of $to_o(S)\}$.

Theorem 2. A set S of clauses is unsatisfiable if and only if there exists an N such that $C_N^{(3)}$ contains the empty chain.

A procedure of the form of the third procedure is called a Model Elimination procedure. If the A-literals are identified with the members of the S-list in [I/I], then this procedure is seen to be essentially that of [LI]. The procedure given here does extend the procedure of [LI] by enabling lemmas to be used in a similar manner to matrix chains to develop new chains by extension. No parallel exists in the method of [Ll].

We consider a simple example contrived to exhibit the use of the three operations of the Model Elimination procedure. The problem is set up by presenting a set S of clauses. The demonstration below establishes that (the closure of) the conjunction of the given clauses is an unsatisfiable wff.

Given clauses: I. Fg(a)x ^ Fg(y)a
II. ~Fg (a)x ^ Fyy
III. -Fax ^ ~Fg(x)y.

IU$_o$ (S) contains six chains, two chains from each clause. In stating the reason for each step of the deduction given below, reference is made to the clause from which the appropriate chain is obtained. Order in the chains below is left to right (i.e. literal L-.$_\perp$ precedes literal L^ in the chain ordering if L$_L$ is to the left of *h*$_?$) .

Deduction (A-literals are underlined)	Reason
1. Fg(a)x Fg(y)a	Initial chain I;
2. Fg(a)x $\underline{\text{Fq(a)a}}$ Fyy	Extension using II;
3. Fg(a)x $\underline{\text{Fq (a)a}}$ $\underline{\text{Faa}}$ ~Fg(a)y	Extension using III;
4. Fg(a)x $\underline{\text{Fq(a) a}}$ $\underline{\text{Fa.a}}$	Reduction (the scope of $\underline{\text{Fq(a)a}}$ is now 1);
5. Fg(a)x	Contraction;

lemmas produced: (1) ~Faa ~Fg(a)a, instance of III;
(2) ~Fg(a)a ;

„ , v   Extension using lemma (2) ;
6. ~~Fg(a)a~~
7. 0 (empty chain)   Contraction;

(lemma produced: ~Fg(a)a, same as Lemma (2)).

The chain of step 5 has a deficit of *2,* the chain of step 6 has a deficit of 1. (The empty chain has a deficit of 2). Therefore there are only two Extension operations on 0 deficit chains so 0eC^$^{(3)}$.

§3. In this section theorems 1 and 2 are established. It suffices to show the $^{?}$only if^{1} statement of theorem 1 for i = 1 and to show the $^{!}$if$^{?}$ statement of theorem 2. Both theorems then follow as each succeeding procedure terminates with the empty chain if the preceding procedure so terminates.

We show first the $^{!}$if statement of theorem 2 which establishes the soundness of the three procedures. This is done by making use of the clash defined within the resolution method (see [R2]). A lemma chain viewed as a clause is shown to be obtained by a suitable clash over previously defined lemmas and perhaps a matrix clause. Using the fact that the use of clashes in resolution leads to a sound procedure, we establish the soundness of the procedures of this paper. Throughout the proof of soundness the lemma is to be viewed as a clause without further comment. That is, the lemma (clause) is the disjunction of the literals which comprise the lemma chain; we will consider the clauses to be a set of literals. These remarks hold also for the matrix clause.

and a substitution 9 such that (0^0,0^9,... ,C^.9) is a clash. C_O is the nucleus of a latent clash. A literal L of C_O is active if $L^!9$ is an active literal of $C^!_O0$.

We now prove a lemma which gives the needed induction step to carry out the proof outlined above. We assume that chain K^1 has been deduced (i.e. there is given a deduction of K^T) and the Contraction operation is being performed on $K^!$. Moreover, this application of the contraction operation may already have removed some A-literals from K^f (and created the corresponding lemma chains). Let K represent the parent chain just prior to the removal of A-literal L,. Thus K is obtained from K^r by removal of the A-literals following L,. in K^f: We are concerned with the lemma clause H created as L_a is removed, By a previous J^emma we mean a lemma (clause) created in any deduction by the procedure prior to this immediate stage in the procedure's development. By a preceding lemma o^f the deduction we mean a lemma (clause) created earlier in the deduction of K^1 or a lemma already created by this application of the Contraction operation.

Lemma JL. Under the assumptions of the above paragraph, H is a clash resolvent whose latent clash has as nucleus a matrix clause or previous lemma and whose remaining clauses are preceding lemmas of the deduction.

Proof; We recall a key property of a lemma. We state the property in terms of H. A literal can appear in lemma H only if it is the complement of an A-literal in the chain K.

As L_A is an A-literal there is an ancestor chain in the deduction of K' to which the Extension operation was applied which resulted in a chain containing literal ${}_A L$, where ${}_A L$ is the earliest ancestor of L_A that is an A-literal. Let ${}_A K$ be the chain containing ${}_A L$. Loosely speaking, it is the point in the deduction where an ancestor of L_A was 'made' an A-literal* The elementary chain C chosen for that application of the Extension operation is the nucleus of the latent clash. The latter statement is well defined only if we regard C as a clause which we now agree to do. It is clearly either a matrix clause or a previous lemma. The literals of C which have descendents in the deduction of K' which are A-literals are the active literals of C. Let $L_1, \ldots, L_r$ denote the active literals of C and $C_{r+1}, \ldots, L_n$ denote the remaining literals of C. There is one clause in the latent clash for for each L_i, $1 \le i \le r$, namely the lemma produced when a descendent of L_i is removed by Contraction. This completes the specification of the latent clash and it remains to show that a clash associated with this latent clash has H as its clash resolvent. We actually show that the clash resolvent R satisfies the key property of a lemma, i.e. that R is a subset of the set S of complements of A-literals of K. We leave to the reader the exercise of convincing himself that the use of the 'scope' mechanism does produce precisely the desired subset of S, namely that $R = H$ given that $R \subseteq S$.

Case 1. We first show there is a substitution *y* such that, for r < i $\leq$ n, L_1.ycS. We in fact will have that *cy* is the nucleus of the desired clash.

If K^o,$K^{\text{┴}}$,...,$K^{\text{ↄ}}$,...,K^m is the given deduction of $K^!$ (thus K' = K_m) there is a j, 0 < j ;< m such that ˋK = $K_{/\!\text{L}}$.$._3$

Because K_{j-1}. and C form the chain K_j. by the Extension (O) operation there must exist a substition 0 such that C9 = C contains only literals appearing in K_j. plus the complement of $_A$L. Let A_V denote the substitution (often null) such that

A K. ˉ ˉ

K_{j+k} = K_{j+k-1}V 1 < ^ < m-ˋ j. Let . C^{*k}> = $C^{<k-L>}$$_V$
1 < k < in - j. Then $C^{\text{™"'}-3'}$ is the nucleus of the clash we seek
and *y* = 0$A_1$$A_2$~...$A_{m-j}$ . . We establish that L_L.yeS, r < i $\leq$ n.

As by lemma hypothesis we are concerned with removing L_A from K by Contraction, all literals of C have already had their descendent literals removed from some appropriate parent chain. In particular all descendents of $L_{.1}$ r < i ≤^ n have been removed. The removal must have been by the Reduction operation or else some descendent of L_1 would have been an A-literal. This implies some A_k made the descendent $L_i^{(k)}$ eC$^{(k)}$ of L_i the complement of a preceding A-literal L* in K. , . We

have L_1y = L|$^{k)}$V l V 2 - V j $^{(or}$ ^{h}$_{i}$^{7} = ^ ' ^ ^{if} ^ = m - j)
complementary to L*$A_{V,1}$...$A_{\text{K.Ij.}}$ $_{\text{m—~2}}$. (or L* if k = m - j). We must show *L** , $_n$$A_n$, $_o$. . . A . is an A-literal of K. Clearly if a descendent of L* is in K it is L*X,,, ...\ $_{\text{K-ri}}$ $_{\text{m—j}}$. (or L* if k = m - j) . Thus the only way the statement could not

be true is if no descendent of L^x is in K. But $C^{(O)}$ minus the complement of L_{Pi} is appended following $L_{r\backslash}$ in $K_{t\backslash} = K_{J}$. Because new A-literals are formed only from last elements of parent chains no new A-literal formed in K^{j+p}, p > 0, can precede any descendent of C in K_{j+p}. Hence L* must be a descendent of $_AL$ or of an A-literal preceding L_A and hence must have a descendent in K. Thus $L_i yeS$, r < i ≤^ n and case 1 is shown.

Case 2. Let H_i denote the lemma formed when a descendent $L_i^{(V)}$ of $L_i eC$, 1 ≤ i ≤ r* is removed by Contraction from some chain K_{j+k}, k ≥ 0. We must show for each i, 1 ≤ i ≤ r, there exists a substitution 0_i such that $H_i 0_i$ has precisely one literal complementing a literal in Cy = C^™"^' and all other literals appear as members of S. (Note that as S contains no pair of complementary literals the remaining condition on defining a clash is satisfied).

The induction hypothesis allows us to assume that H_i is composed of a literal which complements $L_i^{(k)}$ and perhaps complements of A-literals of K_{j+k} preceding $L_i^{(k)}$ in K_{j+k} for a suitable k > 0 . If j + k < m_J, then $0_i = A_{k+1}, A_{k+2}, \ldots, A_{m-j}$. If j + k = m let 0_i be the null substitution. Then $H_j 0_i$ contains a literal which complements $L_i y$. The other literals of H.0. complement A-literals of K. .,0. which contains every literal of K because every literal of K is a descendent of a literal of K. ., . An argument such as for L* in Case 1 shows any A-literal preceding L.y in K . .,0. must be L_ or an A-literal preceding L_A hence an A-literal of K.

As the above proof may be carried out independently for each $L_{i'}$ $1 \leq i \leq r$. Case 2 is complete. It should be noted that (obvious) modifications of the substititions θ_i must be applied to the ^-unconnected variant of the latent clash because of the change of variable names in clauses representing H_i.

The proof of the lemma is complete.

It is quite easy to prescribe an order of development of the third procedure so that the previous lemmas and the preceding lemmas of the deduction precede in development the creation of a given lemma H. (The use of H\(S) as given in the previous section suffices, in particular). Then by induction, supposing earlier lemmas to be obtainable from the matrix clauses by Resolution using clashes, Lemma 1 establishes that H is obtainable from the matrix clauses by Resolution using clashes. Thus every lemma is obtainable from the matrix clauses by Resolution using clashes.

We now show that if the empty chain is derivable by the third procedure of the previous section there is a clash involving matrix clauses and lemmas only whose resolvent is the empty clause. Using the observation of the preceding paragraph, the empty clause is then seen to be derivable from the matrix clauses using Resolution with clashes. Let K_o be the first chain of a deduction of the empty chain. Let C denote the clause containing the same literals as $K_{\bar{o}\wedge}$; thus C is a matrix clause. There is a latent clash with C as nucleus with all literals L_i of C active such that the lemmas H_i associated with each L_i comprise the remaining clauses of the latent clash. The associated clash has the empty clause as clash resolvent.

To establish these last statements note that there can be no A-literal preceding any descendent of a literal of K_o in any chain of a deduction beginning with elementary chain K_o. Hence in a deduction of the empty chain every literal of K_o has a descendent which is an A-literal as the Reduction operation cannot be used to eliminate any of these literals. Thus each L_i of K_o (hence C) has a lemma H_i associated with it created when the descendent was eliminated. Hence each L_i of C is active. Each lemma H_i must be a one-literal clause also by the above observation and the requirement that each literal of H_i must be the complement of an A-literal in a chain that in this instance contains only one A-literal. Hence the clash resolvent must be empty if a clash is possible at all. But the proof of Lemma 1 determines the substitutions needed to form the clash from the latent clash so a clash is possible.

We have established that the empty clause is derivable using Resolution with clashes if the empty chain is derivable. The soundness of Resolution assures us that if the empty clause is derivable the set of matrix clauses is unsatisfiable; thus the same is true if the empty chain is derivable. The ^{f}if* part of theorem 3 is established.

It remains to establish the $^{?}$only if^{1} statement of theorem 1 for i = 1. The *only if^{1} statement requires us to prove that if a set of matrix clauses forms an unsatisfiable set of clauses, then $C\wedge^{'1\backslash'}_{J}$ contains the empty chain for some positive integer N. This property, the <u>completeness</u> property, then follows for all procedures as for any non-negative integer k, ^{c}-^$^{'1)}$ £ ^{c}T!$^{'2)}_{\wedge}$ £ ^{c}v$^{(3)}_{\wedge}$.

We represent the given unsatisfiable set § of matrix clauses by {M_{1}, ...,M_{m}}. We let M_{i} also represent any matrix chain derived from matrix clause M_{i} by a suitable ordering of the literals of M_{i}. Let {C_{1},...,C_{r}} denote the truth-functionally contradictory set of clauses corresponding to §. This set is guaranteed to exist by Herbrand^{1}s theorem. We will also use C_{i} to represent an (elementary) chain determined by a suitable ordering over the clause C_{i}. For each i <_ *r,* C_{i} = $M_{j}0_{i}$ for some matrix clause M_{j}, 1 < j < m^ and a suitable substitution 9_{i}. When this latter property holds we say {jyL_{1},...,M_{m}) <u>generates</u> {C_{1},...,C_{r}}.

We are entirely concerned with deductions within the first procedure where the operations are Basic Extension (BE)$_{9}$ Basic Reduction (BR) and Basic Contraction (BC), and no deficit counter is present. It is convenient to work with a certain type of deduction defined here in terms of the first procedure.

<u>Definition</u>: A <u>ground</u> deduction is a deduction where no substitutions are performed on the chains of the deduction by BE or BR. Hence, any literal appearing in any chain of a ground deduction appears as a literal of the <u>given set</u> of chains comprising $\wedge_{o}(^{9})$.

The term 'given set[1] in the above definition replaces the term ^{f}set of matrix chains* as the initial set of chains, i.e. $C_o^{(1)}$, as we will use other sets than matrix chain sets as starting sets for the first procedure. In particular, we shall use $\{C_1,.,.,C_r\}$. The following lemma in fact allows us to work entirely with ground deductions from the given set $\{C_1,...,C_r\}$.

Lemma 2, Let G = {M^,...,]^) 9enerate B = $(C_{]L},...,C_r)$. Then for every ground deduction G $S_o, _1 . . . ,_n$G from given set B of chains there is a deduction $K_o,...,K_n$ of the first procedure from the given set G of chains and a set $\{A._1,...,A_n\}$ of substitutions such that $K.A._{1\,1} = G._1$ 0 <_ i <^ n •

Let us first observe why this lemma allows us to work simply with ground deductions. If we can show for every contradictory set of clauses there is a ground deduction of the empty chain, then Lemma 2 asserts there is a deduction of the first procedure yielding the empty chain which has as the given set the set of matrix clauses generating the contradictory set. This is the result we seek.

Proof: The proof is by induction on the length n of the deduction,

n = 0. G_O must be a $C._I$eJi. Thus there exists a j and $6._1$ such that $C._l = M._j6._l$, $M._j$eG then K_O is $M._j$ and $A_O = 6._l$.

n = k. We assume the lemma holds for n = k - 1. $G_,^K$ may be derived from $G_,^{K-1}$ using BE, BR or BC. We investigate each possibility in turn.

BE. The operation BE has two input chains, G,K-1 and some C. eB. We have C. = M.0. for some M. rG. Let L,
1 1] 1] 1
denote the last literal of K,- and let L? denote the first
k-1
literal of chain M.. Let A, and A.. be the atoms associated
D 1 Z
with L, and Lo respectively. By the assumption that G,
1 Z K
is a chain of a ground deduction and derived from GV i bY BE>
we have A^. , = A0θ. . Let K, .^ denote the x-instance of
K-1 1 K-M1 Z 1 K—1 J\ ]
K, , and M.7? denote the y-instance of M., then from the
above there exists a substitution *y* such that Kn -,£T,7 = G, ,,
K—1 Jtx K—1
M.T7My = C. and A- *L.y* = Ao77_.y. By the Unification theorem - 'K
] M 1 JL i\ *Z M.*
there exists a most general unifier a such that A,£ or = A^Tha.
K
That is^ Ln *i^o* and LoT]T^a are complementary literals. From the
IK Z JK
definition of BE it follows that K, is composed of chain
Ki *i^ti®* with the chain M, *7) p* appended minus its first literal.
K—1 K KM
By the Unification theorem there exists a substitution A, such that $G_k = K_k A_k$.

BR. Let L.. denote the A-literal and Lo denote the
K-1 K
B-literal in K, n such that Ln Av 1 and L0Av n are comple-
K-1 K—1 1 JC—JL Z K.—1
ments in G, , and L2^k 1 is om:i-tted from G, due to the use of BR on G, , . Let A.. and A2 denote the atoms of L.. and L2 respectively. Then ^*i^ i* = A2^k 1- Thus bY the Unification theorem there exists a most general unifier a such that A..CF = A^a. Then L,a and L^cr are complements in K, ,a. Thus by definition of BR, K, is K, ,a with Lpa removed. By the Unification theorem there exists a substitution A_k such that $G_k = K_k \backslash_k$.

BC. If G_k is derived from G_{k-1} by BC then $K_{,K}$ is directly derivable from K_{v} , as BC uses no matching operation. Thus \ = \ , (or \ may be a restriction of 7\, n , that is, simply 'transform' fewer variables if some variables present in K^{K-1} are absent in K^)•

This completes the proof of the lemma.

We are now able to limit our attention to ground deductions. We must show that for every truth-functionally contradictory set {C^1,...,C^r) (which we assume to be minimal) there exists a ground deduction of the empty chain. We now refer to 'ground deductions' simply as 'deductions'. A deduction $* extends deduction & (or &* is an extension of &) if &, a deduction of length n, comprises the first n chains of J8*. We define a literal L appearing in a chain of deduction & to be live in & if there is no extension &* of & in which a descendent of L can be eliminated by either BR or BC. Clearly a descendent of a live literal is also live. Also a live literal cannot be an A-literal and the last literal of a chain in a deduction for it would be removed by BC.

Let B represent the given minimal contradictory set of clauses {C^...,C^}. Let C^o denote an arbitrary member of B which is taken to be G_o . Call this one-step deduction $\&_o$. (We continue the practice of regarding C. as a clause and also, under an often unnamed ordering, as an elementary chain. When the ordering is important it is specified.) If we can show that no literal of C_o is live in $\&_o$ then there must exist a deduction of the empty chain. For suppose there is no deduction & such that * is an extension of. $\&_Q$ and & derives the empty chain. Then the first literal L of C_Q must be live in fi_Q because its descendent L^f could only be removed by BC which can only remove last literals of chains. Hence removal of L^1 by BC must leave the empty chain as derived chain. Thus we need show only that no literal of C_o is live in $\&_Q$.

For convenience we will hereafter omit the terms !descendent^1 and 'ancestor^1 of a literal L in a deduction \$ and refer simply to L. With no substitutions occurring in (ground) deductions the descendents and ancestors of L are the same literal as L. Thus we shall say 'L is never removed in &\ for ^{f}a descendent of L is never removed in #!, for example.

We shall assume that at least one literal of C_o is a live literal in $\$^o$ and produce a contradiction with the fact that B is a minimal contradictory set. We use in the argument chains whose last literal is live in the deduction in which the chain appears. In particular, we see below that when we apply BE to such a chain we must introduce a new live literal (in the extended deduction). A chain whose last literal is live (in the deduction in which it appears; if an isolated chain then in the deduction in which it is sole chain) is called a primary chain. We have the following lemma.

~~Lemma 3~~_. Given a deduction \$ of a chain K we can extend \$ to a deduction \$! of chain K^1 where K! is either a primary chain or the empty set. Moreover,, every literal of K! is a literal of K and the ordering of K holds in K^T (i.e. $K^?$ is a ~~subchain~~ of K).

Proof: Let L be the last live literal of K and let $L_j,\ldots,L_s$ denote the (non-live) literals following L. (if K contains no live literal let $L_j,\ldots,L_s$ denote the literals of K.) K' will be K minus all literals following L (i.e. K^1 is the ~~initial subchain~~ of K with last literal L). We define the extension $' of fi such that there exist chains $K_0, K_1, K_2, \ldots, K_s$ where $K_0 = K$ and $K_s = K'$, K_{i+1} is a subchain of K_i with one of $L_1,\ldots,L_s$ removed, and K_{i+1} follows K_1 as a chain in $' . It suffices to show how to obtain K_{i+1} from K_i. There are two cases, one each for removal by BR and BC.

Removal by BR. For this case K_{i+1} follow K_i as next chain in &? . The appropriate literal L_j, $1 \leq j \leq s$, is removed by BR due to preceding complementary A-literal. All possible removals by BR are assumed performed prior to a removal by BC. That is, there exists a t, $1 \leq t \leq s$, such that $i < t$ implies K_{i+1} comes from K_i by BR, $i \geq t$ implies K_{i+1} comes from K_i by BC.

Removal by BC. K_{t+1} from K_t is the first case of removal by BC • If t = s, no removal by BC is necessary. Otherwise, the last literal of K_t is not live hence removable by BC. Append the deduction which removes this literal. K_{t+1} is the last chain of this deduction, K_{t+1} must be an initial subchain of K_t because BC only removes last literals of a chain. It is possible to remove several of the L_j's on one operation if literals just preceding the last literal are already A-literals.

Then K_{i+1} coincides with K^ in which case we define K_{i+1} as following K_i in #$^!$. Each last literal of *K^,* t <_ i - s, can be removed in this fashion so K_s is deduced by fi^1 'as prescribed.

This completes the proof of the lemma.

Given a deduction $ of K we denote by &* and K* the deduction and primary chain respectively obtained by means of Lemma 3. Thus #* is an extension of &. If we apply BE to primary chain K (having deduction &) and let $K,^1$ denote the derived chain of BE and $\#.^1$ the associated deduction then K^1 contains K as a proper initial subchain. This means there is a new live literal following the last literal of K within K^1 . For suppose these assertions false. Note the manner of the proof of Lemma 3 results in no removal of literals before the last live literal so all literals of K in $K.^1$ remain in Kt^1. Hence K^1 is an initial subchain of K. BE makes the last literal L of K an A-literal. If L is the last live literal in $K,^1$, then by Lemma 3, K^{1*} = K. But a primary chain cannot have its last literal an A-literal because BC can remove the literal contradicting the fact it is live. The assertions are thus seen to be true.

Let BE* denote the operation which takes primary chain K to primary chain K^1 as above thus extending deduction $ to deduction $\$^1$. We develop a collection of (ground) deductions whose existence is shown to contradict the minimality of B. We need only the operation BE* (which of course employs BE, BR and BC) .

We start with the basic assumption: C_o contains a live literal in deduction $\$_o$. We extend $\$_o$ to deduce C^*_o, a non-empty primary chain. We define two lists we maintain which determine which deductions we develop. One list^ the L-list, will consist of all the live literals which appear as last literals of primary chains derived using BE*, and also the last literal of C^*_o. This list will include all A-literals of the deductions we form. No literal entered on the L-list is ever removed. The other list is the ~~clause list~~ initially containing the clauses of (B . Every time a literal L_1 is added to the L-list,, all clauses containing L_1 are removed from the clause list (and never replaced). When BE* is applied to a primary chain, the first step of its execution is an application of BE. The elementary chain used by BE must be formed from a clause remaining on the clause list.

We define the list $1_1, 1_2, , ,_M J5_t$ of deductions* $\&_1$ is determined as follows. We have deduction $\$^!$ of C^*_o. The last literal of C^{φ} is placed on the L-list as required and all clauses containing that literal are removed from the clause list. Now apply BE* to C^{φ}. The resulting primary chain has its last literal placed on the L-list and the clauses containing that literal removed from the clause list. This continues until BE* can no longer be applied. This occurs when it is impossible to form an elementary chain from a remaining clause of the clause list such that the operation BE can be applied. When the deduction cannot be extended by an application of BE*. we say the deduction has terminated. The terminated deduction is fi_{1}. The last literal L of the last chain of $\$,_{1}$ (now a member of the L-list) has the property that no literal identical or complementary to L occurs in a clause of the clause list. No identical literal occurs as all clauses containing L are removed when L is added to the L-list. No complementary literal occurs as $\$._{1}$ is a terminated deduction. Such a literal on the L-list is said to be closed. Deduction $*\underset{1}{-}{}^{+}\underset{1}{i}$ ^{is} formed from deduction $\&._{1}$ by taking as given as large an initial part of $\&._{1}$ as allows an extension to be made by BE*. The deduction is extended until termination. The terminated deduction is $\wedge-\underset{1\mp 1}{.-i-}$ Note that any subsequent extension of any initial part of $\$._{I}$ by BE* cannot be an initial part of $\$._{i}.$ For the chain used by BE* originally in developing $\$._{1}$ is no longer obtainable as the corresponding clause is not in the clause list. Thus $\&_{l+l'}.$ is distinct from $\$_{1}..$ One may view the construction of $\$_{1+1}$ from $\$_{I}$ as removing chains from the end of the deduction $\&_{1}$ until one first encounters a primary chain that can be extended by BE* because a complementary literal exists in some remaining clause of the clause list. Then extend (the deduction) by BE* until termination. In the !backing up^{T} process to find the appropriate initial part of $\&_{1}$ all literals on the L-list whose chains were dropped are now closed literals. That is, all literals on the L-list whose chains (of which they are last literal) are in $\$_{I}$ but not in $\$_{I+1}$ are closed.

Because the clauses of the initial clause list (i.e. g) are finite$_9$ only finitely many $\&_{r_K}$ can be formed in the above manner, i.e. t is a positive integer. After $\&_t$ is defined, all literals of the L-list are closed. We observe first that the literals of the L-list form a consistent set of literals. We show no literal shares the same atom with a following entry on the L-list. Let L be a given literal of the L-list. A literal following L is either a live literal on a chain containing L as a preceding A-literal or is entered on the L-list after L is closed. In each case the possibility of sharing the same atom is ruled out. Thus the L-list literals form a consistent set. Thus the collection of clauses removed from the clause list forms a consistent conjunction of clauses. Hence,, the clause list must still be non-empty. Moreover^ it must contain a contradictory set of clauses. This is true because no literal of a clause of the clause list shares an atom with a member of the L-list (as all L-list literals are closed) . But the remaining clauses of the clause list is a proper subset of (B (as at least C_o is missing) . This contradicts the fact that B is a minimal contradictory set of clauses. Hence no literal of C_o is live.

This establishes the 'only if^1 statement of theorem 1, for i = 1. Thus all theorems are now proven.

We should note why it is sufficient to consider only preadmissible chains in building deductions. The reason of course is that only these chains are used in building the deductions needed to obtain a contradictory set smaller than B. There are three restrictions on a preadmissible clause. Part a) follows as B-literals between successive A-literals come from the same clause and no tautological clause is needed in a contradictory set of clauses. Part b) and one half of part c) occur because BE uses only chains (clauses) from the clause list devoid of clauses containing literals which are also A-literals. The remaining half of part c) follows as BR could remove a complementary literal before it becomes an A~literal.

A final word concerning the procedures and the completeness proof. One might regard the philosophy of these procedures as 'attempt to show B consistent and show this attempt must fail'. (The interest in a procedure of this type is that it wastes minimal time on its mistakes, the consistent sets it 'constructs' seeking B). Making a literal an A-literal may be regarded as fixing a truth assignment for an atom of B. (We regard A-literals as assigned the truth value T. The atoms are in effect statement letters of a propositional statement corresponding to B.) If a sufficient number of atoms of B are given truth assignments, a _false clause_ (a clause with all literals receiving truth assignment F) must appear. The essence of the completeness proof is to show we can always define a sufficient number of A-literals to force a false clause under constraint of linking clauses together by complementary literals. That is, we had to establish that a string of clauses exists that links an arbitrary clause of ft with a false clause. (The false clause is defined, of course, only in the process of developing the string of clauses itself). Once such a linkage is completed (which results in a preadmissible chain) all !models^f for B sharing that partial assignment of truth values to atoms of B can be eliminated as ^{1}proofs^1 of consistency of 6. Hence the name Model Elimination. Forcing the linking condition on the string of clauses connecting an initial clause with a false clause forces considerable structure on the collection of clauses containing the false clause. Often quite a bit about B is known when the first (non-trivial) false clause is found. It is this last fact which makes useful the notion of !partially contradictory! set of clauses (utilizing the device of !positive deficit^1 clauses).

References

[Ll] Loveland, D. W., ^{f}Mechanical Theorem-Proving by Model Elimination^1 Jjour. ACM (to appear) .

[Rl] Robinson, J. A., !A Machine-Oriented Logic Based on the Resolution Principle', Jour. ACM, 12 (Jan. 1965), 23-41.

[R2]_________________, *A Review of Automatic Theorem-Proving^1, Proc. Symposia in Applied Math. (American Math. Soc.), XIX, (1966).

Carnegie-Mellon University
Pittsburgh, Pennsylvania