AN EQUATIONAL APPROACH TO THEOREM PROVING IN FIRST-ORDER PREDICATE CALCULUS

Deepak Kapur* and Paliath Narendran

Computer Science Branch
Corporate Research and Development
General Electric Company
Schenectady, New York

ABSTRACT

A new approach for proving theorems in first-order predicate calculus is developed based on term rewriting and polynomial simplification methods. A formula is translated into an equivalent set of formulae expressed in terms of 'true', 'false', 'exclusive-or', and 'and' by analyzing the semantics of its top-level operator. In this representation, formulae are polynomials over atomic formulae with 'and' as multiplication and 'exclusive-or' as addition, and they can be manipulated just like polynomials using familiar rules of multiplication and addition.

Polynomials representing a formula are converted into rewrite rules which are used to simplify polynomials. New rules are generated by overlapping polynomials using a critical-pair completion procedure closely related to the Knuth- Bendix procedure. This process is repeated until a contradiction is reached or it is no longer possible to generate new rules. It is shown that resolution is subsumed by this method.

Key Words: Theorem Proving, Automated Reasoning, Equational Approach, Term Rewriting, Knuth-Bendix Completion Procedure, Polynomial Simplification.

1. INTRODUCTION

A new approach for proving theorems in first-order predicate calculus is presented. The approach is based on term rewriting and polynomial simplification methods, and is simple to understand. A key idea is the observation that formulae can be viewed as polynomials over atomic formulae when they are expressed solely in terms of boolean connectives 'exclusive-or', 'and' and constants 1 and 0 which stand for truth and falsity, respectively. In this representation, formulae can be manipulated just like polynomials using familier rules of multiplication and addition; the addition ('+') is 'exclusive-or' and multiplication ('*') is 'and.' Further, the polynomials we encounter are simple because we neither see coefficients other than 1 nor degrees more than 1; these polynomials satisfy the additional properties that for any polynomial p, $p + p = 0$ and $p * p = p$. The method works well irrespective of whether the input formula has a clausal or a non-clausal representation (Chang and Lee, 1973). For applications of theorem proving in artificial intelligence, program verification and synthesis, specification analysis, etc , an interested reader may wish to look at (Chang and Lee, 1973; Robinson, 1965; Slagle, 1974).

In our method, if a formula is to be proved valid (unsatisfiable, respectively), it is asserted to be 0 (1, respectively), and a contradiction is derived. Towards this end, an equivalent set of formulae (polynomials) expressed using 'exclusive-or,' and,' and 'true', are generated from this assertion. This is done using the natural deduction approach by analyzing the semantics of the top-level operator. It is checked whether a contradiction 1 = 0 can be derived from these polynomials. One good way of checking for a contradiction is to use the rewriting concepts to generate a Grobner basis of the original set of polynomials by suitably modifying the method developed in (Kandri-Rody and Kapur, 1984) for computing the Grobner basis of an ideal over polynomial rings over the integers. Checking whether a contradiction is derivable from a set of polynomials is equivalent to checking whether their Grobner basis is trivial in the sense it includes 1.

For the propositional calculus, this approach is a

Partially supported by the National Science Foundation grant MCS-82-11621.

straightforward application of the Grobner basis, algorithm developed in (Kandri-Rody and Kapur, 1084). For first-order predicate calculus, additional techniques are developed based on identifying equivalences among formulae using unification. Polynomials are first transformed into rewrite rules and the Grobner basis is computed by generating critical-pairs among rewrite rules The critical-pair generation is similar to resolution but is more powerful and allows a lot more of flexibility; in fact, it is shown that it *subsumes* resolution. New polynomials are generated from the critical pairs to augment the basis set of polynomials to obtain a Grobner basis. If the Grobner basis consists merely of the rule 1 — 0 then we know that the original formula is unsatisfiable or valid depending upon what we were contradicting.

This approach is particularly useful for reasoning about domains which can be axiomatized using a finite set of equations, Term rewriting approach for developing decision procedures for equational theories can be integrated well with lirst-order predicate calculus as first-order predicate calculus itself can be handled using rewrite rules.

The proposed approach is motivated by Hsiang's method for theorem proving in first-order predicate calculus based on term rewriting (Ilsiang and Dershowitz, 1983) and the approach for generating a Grobner basis of a polynomial ideal over the integers in (Kandri-Rody and Kapur, 1981). At the theoretical level, the major distinction between our approach and Hsiang's method is that our method is based on the Grobner basis computation whereas Hsiang's method is based on the extensions of the Knuth and Bendix completion procedure for handling associative and commutative operators developed in (Lankford and Ballantyne, 1977; Peterson and Stickel, 1981). Further, we believe our definitions of unification of monomials, rewriting, superposition and critical pairs are conceptually simpler to understand than Hsiang's. At the implementation level, our approach seems to be easier to implement than Hsiang's, as it does not need to use associative-commutative unification algorithm or its variation, called BN-unification by Hsiang, to handle boolean operators. An implementation of our method is underway. At this stage, it is difficult to make any comparison between the running times of Hsiang's method and our method; however, initial results seem to suggest that our method is more efficient than Hsiang's. Further, it is too early to say how our method compares with resolution or other theorem proving methods. However, in case of propositional calculus, preliminary experiments suggest that our method is more efficient than the resolution-based LMA theorem prover.

In this paper, we give an overview of the approach exhibiting how first-order formulae can be viewed as polynomials and rewrite rules. The method is illustrated using examples. Further technical details and proofs of the theorems in this paper are given in (Kapur and Narendran, 1984).

2. PROPOSITIONAL CALCULUS

Given a formula / , it is asserted to be I or o depending upon whether / is to be shown unsatisfiable or valid, respectively, and it is checked whether a contradiction can be derived. In order to do so, the resulting equation is first translated into an equivalent set of polynomials expressed using 'exclusive-or' (+), 'and' (*), 'true' (I) and 'false' (0). (Since $a + a$ — 0 as well as a - a = 0, we use ' + ' and '-' interchangably in the paper; we also often omit '*' among atomic formulae.) This can be done by analyzing the outermost operator of the left-hand-side of the equation; see (Kapur and Narendran, 1981) for details. In many cases, we may able to generate the contradiction 1 = () in the process of obtaining an equivalent set of polynomials, in which case we are done. If not, then the translation gives a set of polynomial equations, say {p, = 0 | 1 < i < n }.

To check whether the polynomial equations lead to a contradiction or not, we generate its Grobner basis (Buchberger and Loos, 1982; Kandri-Rody and Kapur, 1984). Informally, a finite set of polynomials constitute a Grobner basis if and only if evey polynomial has a unique normal form when simplified or reduced using polynomials in the basis. To generate a Grobner basis, each polynomial equation is converted into a *rewrite rule.* This is done by totally ordering atomic formulae which is always possible in the case of propositional calculus. This ordering is extended to *monomials* (products of atomic formulae) based on degree (i.e., size) and lexicographic ordering (cf. Buchberger and Loos, 1982; Kandri-Rody and Kapur, 1984). (In fact any ordering which satisfies the following properties will do: (a) 0 < I < m for any monomial m different from 1 and 0. (b) $m_1 < m_2 \Rightarrow s\, m_1 < s\, m_2$ for every monomial S.) Under a total ordering on monomials, every polynomial has a *unique head-monomial,* which serves as the *left-hand-side* of its rule; the *rest* of the polynomial serves as the *right-hand-side* of the rule.

We also include a rule A',"' -> A', for each propositional variable A, *(idempotency rules)* as well as another rule 2 -+ 0. Thus we do not have to deal with any indeterminate of degree more than 1 nor do we have to deal with coefficients other than 0 or 1. There is also a rule *P* + 0 -- />, but this is taken care of automatically in the Grobner basis computation.

The rules thus obtained are used to rewrite polynomials. From these rules, new rules are generated to look for a contradiction. For each pair of distinct rules, the overlap of their left-hand-sides is generated by taking the least common multiple (1cm) of the left-hand-sides. Consider two distinct rules *L*, -♦ R_x and L_2-+ R_2. Let /. be the lcm of L_x and A_y, so *I* =- F_x L, = F_2 L_2.

$$\begin{tikzcd} & L \arrow[dl, "L_1 \to R_1"'] \arrow[dr, "L_2 \to R_2"] & \\ F_1 R_1 & & F_2 R_2 \end{tikzcd}$$

Then <F, R_u F_2R_2> is a *critical pair* generated by the two rules, and F_XR_X F_2R_2 is its *S-polynomial* (Buchberger and Loos, 1U82; Kandri-Rody and Kapur, 1Q84). If the two polynomials in a critical pair do not reduce to the same normal form (or equivalently, the corresponding S-polj nomial does not reduce to 0), a new rule is added from the normal forms thus obtained. This process is repeated antil no new rules are generated. The resulting basis is a Grobner basis of the input polynomials. For proofs of correctness and termination of this algorithm, see (Kandri-Rody and Kapur, 1684) as this algorithm is a special case of the algorithm for generating the Grobner basis of an ideal over polynomial rings over the integers.

We would like to point out that in order to deduce a contradiction from a finite set of polynomials, it is not necessary to consider superpositions with the idempotency rules; however, these superpositions are needed if a Grobner basis is to be generated. In many cases, generating superpositions with idempotency rules gives rise to simpler rules which perform considerable reduction.

EXAMPLE: Consider the problem of checking whether the following propositional formula / is unsatisfiable.

$$\begin{aligned} f \equiv & [\neg((x_1x_2 \equiv x_1y_2) \equiv (x_2y_1 \equiv y_1y_2))] \\ & \wedge [\neg(((\neg((x_2x_3 \equiv x_3y_2) \equiv x_2y_3)) \equiv y_2y_3))] \\ & \wedge [(x_1x_3 \equiv x_1y_3) \equiv \neg((x_3y_1 \equiv y_1y_3))]. \end{aligned}$$

To prove the formula to be unsatisfiable, we equate / to 1. Using the semantics of A and translating s into +, we obtain the following polynomials:

$$\begin{aligned} & x_1x_2 + x_1y_2 + x_2y_1 + y_1y_2 = 1, \\ & x_2x_3 + x_3y_2 + x_2y_3 + y_2y_3 = 0, \\ & x_1x_3 + x_1y_3 + x_3y_1 + y_1y_3 = 1. \end{aligned}$$

Transforming these into rewrite rules using the ordering $x_1 > x_2 > x_3 > y_1 > y_2 > y_3$ on propositional variables which induces an ordering on products of propositional variables based on size and lexicographic ordering, we get

$$\begin{aligned} & 1.\ x_1x_2 \to x_1y_2 + x_2y_1 + y_1y_2 + 1 \\ & 2.\ x_2x_3 \to x_3y_2 + x_2y_3 + y_2y_3 \\ & 3.\ x_1x_3 \to x_1y_3 + x_3y_1 + y_1y_3 + 1 \end{aligned}$$

We cannot use these rules to reduce each other. So, we generate new rules by superposing these rules on each other. Overlapping rules 1 and 2, we obtain a product *! x_2 *« on which rules 1 and 2 can be applied.

$$\begin{tikzcd} & x_1 x_2 x_3 \arrow[dl] \arrow[dr] & \\ x_1x_3y_2 + x_2x_3y_1 + x_3y_1y_2 + x_3 & & x_1x_3y_2 + x_1x_2y_3 + x_1y_2y_3 \end{tikzcd}$$

If we further reduce these polynomials using the above rules, we obtain $*_8$ on one side and y_a on the other side and all other products cancel with each other. This gives us a new polynomial $*_a$+ y_s = 0. The new rule for this polynomial is: 4. x_s — y_8. Now rules 3 and 4 give us the contradiction 1 — 0.

There is a much simpler way of doing the above example by factoring the above polynomials; see (Kapur and Narendran, 1984) for details. Formulae involving many equivalence connectives are known to often give a lot of trouble to theorem-provers based on resolution, natural deduction and semantic trees. In the case of propositional calculus, they can be easily taken care of in our approach because 'exclusive-or' is one of the main operators.

2.1 Theoretical Foundations

Formulae in polynomial form, as stated above, are elements of a polynomial ring over a Boolean ring, *B* = ({o, l}, +, *). Let *B\X*$_{lf}$*...,X*$_n$\ denote the ring of polynomials over B with the additional property that A', *A, = A, for each A',. Thus if A',, ..., A'„ are the atomic formulae (propositional variables) in / , then / is an element of $B[X_1....,X_n\$.

Theorem 2.1: Let / be a propositional formula with propositional variables A'$_1$ ... ,X_n. Let / be the set of all polynomials in $B\X_{It}$ ... ,$X_n]$ which evaluate to 0 for all assignments of A'$_1$ ..., A'„ on which / evaluates to 0. Then / is an ideal of $B\X_1$ ... ,$X_{n]}$.

Henceforth we refer to the ideal / mentioned in the above theorem as 'the ideal generated by / \ One way to check whether a formula / is valid or unsatisfiable is to analyze the ideal generated by / .

Theorem 2.2: A propositional formula / is valid if and only if the ideal generated by / over $B\X_{X,}$. .X„), wheTe the A,s are propositional variables in / , is trivial (i.e., the whole polynomial ring).

An analogous theorem for an unsatisfiable formula / is that the ideal generating by / +1 is trivial. The triviality of an ideal can be checked by generating its Grobner basis. If the Grobner basis of an ideal contains 1, then it is trivial. The method discussed in the previous subsection for testing validity or unsatisfiability is based on Theorem 2.2 and the Grbner-basis-based test for triviality of an ideal.

3. FIRST-ORDER PREDICATE CALCULUS

The approach discussed in the previous section extends to first-order predicate calculus. The following observations are crucial in working out this extension. Firstly, the atomic formulae in first-order predicate calculus play the role of propositional variables. Secondly, quantifiers can be handled by the method of introducing Skolem functions, known as 'Skolemization' (Chang and Lee, 1973). Thirdly, there are in general infinitely many atomic formulae; however unlike in the case of propositional calculus where propositional variables are independent and unrelated to each other, atomic formulae in first-order predicate calculus are related through the mechanism of *substitution* for variables. Intuitively, this captures the forall rule: *\{\ix) A(x)\ => A{t),* where appropriate restrictions are placed on the term *t* which is substituted for an occurrence of *x* (cf. Chang and Lee, 1973). Another technical issue is that in general, it may not be possible to totally order atomic formulae in the case of predicate calculus (again unlike the case of the propositional calculus); by relaxing this requirement, we also allow more flexibility as well as obtain more general results. The rule corresponding to a polynomial form of a first-order formula thus may have more than one monomial on its left-hand-side. As we show later, if a formula is represented in clausal form (i.e., *CNF),* then every polynomial in the set of polynomials equivalent to the formula has a unique head-monomial; furthermore, it is also sufficient to consider those critical pairs which lead to polynomials with a unique head-monomial.

We first briefly discuss how to handle quantifiers. If the top-level operator of a formula / = 1 is a V quantifier and the associated variable is z,, then remove the quantifier. If the top-level operator is a "1 quantifier and associated variable is *x,*, then introduce a Skolem function .v,, and replace the occurrences of x, bound to this quantifier by 5, (x$_1$.....**), where x_h ..., x_k are the free variables in / . Similarly, we have the dual case for the equation / -a o, i.e., we Skolemize \f quantifiers and remove E quantifiers. We do not need to bring the formula / into prenex normal form first for Skolemization. Instead we Skolemize the quantifiers in place after performing miniseoping so that each Skolem function depends upon as few number of free variables as possible (Bledsoe and Tyson, 1978). This is done by assigning signs with each of the quantifiers and formulae, and giving rules about how the signs change under various boolean connectives; see (Bledsoe and Tyson, 1978) for details.

3.1 DERIVING A CONTRADICTION

Assume that we have obtained a set of polynomials following Skolemization from the original formula without encountering a contradiction; we call this set a *basis* of the formula. Like in the case of propositional calculus, we generate new polynomials from those in the basis using the method of critical pairs and add them to form a new basis. This completion process (which is related to the Knuth-Bendix completion procedure (Knuth and Bendix, 1970)) is continued until it is no longer possible to obtain new rules; In the case of first-order predicate calculus also, we abuse the terminology and call the resulting basis a Grobner basis. As discussed in (Kapur and Narendran, 1984), the Grobner basis of a finite set of first-order formulae has the property that every polynomial in their first-order ideal reduces to 0; however, it does not satisfy the other property that every first-order polynomial has a *unique* normal form with respect to the basis. For propositional calculus, this procedure for generating a Grobner basis is guaranteed to terminate. However, for first-order polynomials, the process of generation of new polynomials may never terminate in some cases. Thus a Grobner basis in the first-order case may be infinite.

The theoretical basis of the Grobner basis approach for first-order predicate calculus is an extension of the results for propositional calculus discussed in Section 2; for details, see (Kapur and Narendran, 1984). We introduce there the notion of a *first-order ring* of polynomials which generalizes the concept of a boolean ring of polynomials, the algebraic structure embodying propositional calculus. We also define a *first-order ideal* to characterize first-order inference in an equational way. This gives us a new way to study first-order predicate calculus in terms of equational logic.

We also show in (Kapur and Narendran, 108-1) that the Cirobner basis approach is *refutation-complete,* or, in other words, if a formula is unsatisfiable, then the Grobner basis computation will terminate with the rule I — o.

3.2 FIRST-ORDER FORMULAE AS REWRITE RULES

A first-order polynomial *P* can be represented as a multiset of monomials (which is a conjunction of atomic formulae*) in *P*; a monomial other than 0 or 1 is assumed not to contain 0 or 1. Let *SAf(P)* denote the multiset of monomials in *P*. A monomial is represented as a multiset of atomic formulae.[1] Given two first-order polynomials ^{f}\, *P*, P* i^{s} *included* in P_2, written as /', C P_2, if and only if $SM[P_X)$ is a subset of $\$Af\{P_2)$.

Two monomials A/, and M_2 are said to be *unifiable* if and only if there is a substitution *a* such that *CT{MJ) = o[M₂)* when considered as multisets. Monomials A/, and $A/_2$ *overlap* if and only if there is a substitution *a* such that *cr{M$_{V)}$* and *cr{A1$_2$)* have a non-trivial greatest common divisor (*gcd*); i.e., *a* unifies at least one atomic formula each from M_x and $A/_2$ (in other words, the intersection of *(T(M\)* and <7(A/$_2$), viewed as multisets, is non-empty).

3.2.1 Partial Weil-Founded Orderings on Monomials

Let < be a well-founded simplification ordering of atomic formulae and terms that is closed under substitutions. An example of a class of such orderings is the 're-cursive path ordering' scheme of Dershowitz (Dershowitz, 1082). The ordering < can be extended to a well-founded ordering « on monomials using the multiset ordering given in (Dershowitz, 1982). The ordering « on monomials extends naturally to a partial ordering on polynomials considered as multisets of monomials.

Using the ordering « on monomials, we can define *head-monomials,* denoted by *HI)(P),* of a polynomial /' as the set of maximal monomials in *P.* In general, *HD{P)* can have more than one monomial. Let *TL* (/') = *P - III) (P*). Details of an ordering on atomic formulae are discussed in (Kapur and Narendran, 1984).

3.2.2 Rewrite Relation

The rule corresponding to a polynomial /' is *HI)[P) — TL{P).* The rewrite relation -> induced by a rule is defined as follows: a polynomial *Q* can be rewritten using a rule *hd* -*■ *tl* if and only if there exist a monomial *m* and substitution *B* such that *m * e{hd)C Q.* We replace m * *0{hd)* by m * *6{tl)* and get *Q' =* [Q m * 0{hd)) \j m * \${tl)* and say that *Q — Q'* by the rule *hd — tl.*

As in the case of propositional calculus, the following rules are also used for rewriting in addition to the rules corresponding to polynomials:

(#,) *p* +0-»p, where *p* is a variable ranging over polynomials; this rule is built into the Grobner basis computation.

(D$_2$) for every n-ary predicate symbol *P:* $P(x_1....x_n)$ * $P(x_l\ \ .,x_n)$ - $P(x_{lt}...,x_n)$.

(B_3) 1 + 1-0.

The reflexive, transitive closure of — is denoted by —'. As is usually done in the literature, we often say *P reduces to Q* if *P* -► * *Q* . It should not be hard to see that for any finite set R of polynomials, the rewrite relation induced by H is Noetherian (i.e. the rewriting process terminates).

1 . Before a polynomial is rewritten using a rule, it įs always assumed to he flat. The result of rewriting however may produce a polynomial that is not flat. That is the reason for viewing a monomial as a multiset of atomic formulae and a polynomial as a multiset of monomials Heneeforth, operations U, fI, etc., are on multisets.

3.3 CRITICAL PAIRS AND GROBNER BASIS COMPUTATION

To check for contradiction, we generate new rules by superposing the existing rules in a given basis. We first consider a simple case; later we consider a more general case.

I. Given two rules (not necessarily distinct)

1. $hd_1 \rightarrow tl_1$ and
2. $hd_2 \rightarrow tl_2$

in a basis, where hd_1 and hd_2 are monomials, if hd_1 and hd_2 overlap, then a superposition P is constructed as follows:

Let σ be a most general unifier unifying G_1 and G_2, where G_i is a subset of hd_i, $i = 1,2$. Let $\sigma(G_1) = \sigma(G_2) = G$, say, and $G_i\ f_i = hd_i$, $i = 1,2$. For each such G_1, G_2, and σ,
$P = \sigma(f_1)\ G\ \sigma(f_2) = \sigma(hd_1)\,\sigma(f_2) = \sigma(hd_2)\,\sigma(f_1)$
Then a critical pair is obtained by rewriting P in two different ways:

$$\begin{tikzcd}
 & P \arrow[dl] \arrow[dr] & \\
\sigma(tl_1)\,\sigma(f_2) & & \sigma(tl_2)\,\sigma(f_1).
\end{tikzcd}$$

Now, we generalize the above construction to the case when the left-hand-sides of rules are not necessarily single monomials. We have:
$hd_1 = h_{i_1} + h_{i_2} + \cdots + h_{i_k}$, and $hd_2 = h_{j_1} + h_{j_2} + \cdots + h_{j_n}$
a superposition P is constructed as follows:

If $\exists\, u, v : h_{i_u} = f_1\, G_1$ and $h_{j_v} = f_2\, G_2$ (G_1 and G_2 could be 1), such that G_1 and G_2 can be unified by a most general unifier σ, i.e., $\sigma(G_1) = \sigma(G_2) = G$, say, then for each such u, v, G_1, G_2, and σ, we have $m_1 = \sigma(f_2)$, $m_2 = \sigma(f_1)$ and

$$P = m_1\,\sigma(hd_1) + m_2 \sum_{\substack{l=1 \\ l \neq v}}^{n} \sigma(h_{j_l})$$

$$= m_1 \sum_{\substack{s=1 \\ s \neq u}}^{k} \sigma(h_{i_s}) + m_2\,\sigma(hd_2)$$

Note that $m_1\,\sigma(h_{i_u}) = \sigma(f_1 f_2)\,\sigma(G) = m_2\,\sigma(h_{j_v})$. We have:

$$\begin{tikzcd}
 & P \arrow[dl] \arrow[dr] & \\
m_1\sigma(tl_1) + m_2 \sum_{\substack{l=1 \\ l \neq v}}^{n} \sigma(h_{j_l}) & & m_1 \sum_{\substack{s=1 \\ s \neq u}}^{k} \sigma(h_{i_s}) + m_2\sigma(tl_2)
\end{tikzcd}$$

Thus a critical pair for the above two rules is
$< m_1\sigma(tl_1) + m_2 \sum_{\substack{l=1 \\ l \neq v}}^{n} \sigma(h_{j_l}),\ m_1 \sum_{\substack{s=1 \\ s \neq u}}^{k} \sigma(h_{i_s}) + m_2\,\sigma(tl_2) >$.

If a critical pair $<c_1, c_2>$ is *trivial*, i.e., the S-polynomial $c_1 + c_2$ reduces to 0, then it is discarded; otherwise, the basis is augmented by adding the rule corresponding to a normal form of $c_1 - c_2$.

Note that in general there could be many critical pairs generated by two rules as, firstly, the left-hand-sides of rules are polynomials instead of monomials, and secondly, a monomial could overlap with another monomial in more than one way. And, a rule can overlap with itself and generate a new rule in the first-order case, whereas that does not happen in propositional calculus.

It should be easy to see that the rules (B_1), (B_2) and (B_3) by themselves do not generate any new rules. However, these rules (especially the idempotency rule (B_2) and the "self-inverse" rule (B_3)) can interact with rules in a basis of a first-order formula to generate new rules; the critical pairs among them are defined below.

II. Given a rule $hd \rightarrow tl$, where $hd = h_1 + \cdots + h_k$, and an idempotent rule of the form $P(x_1, \ldots, x_n) * P(x_1, \ldots, x_n) \rightarrow P(x_1, \ldots, x_n)$, such that the predicate P is used in some h_i, the following two cases can arise:
(a) There is a superposition of the two rules: let σ be a substitution which unifies $P(x_1, \ldots, x_n)$ and the atomic formula $P(t_1, \ldots, t_n)$ in h_i, then the superposition S is $\sigma(P(x_1, \ldots, x_n)\, h_1) + \cdots + \sigma(P(x_1, \ldots, x_n)\, h_k)$, where $\sigma = \{ x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n \}$, and the critical pair $<c_1, c_2>$ is obtained from S by respectively applying the two rules.
(b) For the above two rules, i.e., $hd \rightarrow tl$ and $P(x_1,\ldots,x_n) * P(x_1,\ldots,x_n) \rightarrow P(x_1,\ldots,x_n)$, there is yet another way of generating a superposition. If h_i contains two distinct occurrences of the predicate symbol P, say $P(t_1, \ldots, t_n)$ and $P(s_1, \ldots, s_n)$, and σ is a most general substitution which unifies these atomic formulae, then a superposition is obtained by applying σ on hd and a critical pair is obtained by applying the above two rules to the superposed term.

III. The interaction between the self-inverse rule $1 + 1 \rightarrow 0$ and a rule $hd \rightarrow tl$ in a basis results in new rules as follows: If hd contains h_i and h_j which can be unified, i.e., there is a most general substitution σ so that $\sigma(h_i) = \sigma(h_j)$, then the superposition is obtained by applying σ on hd and a critical pair is obtained by applying the two rules to it. These superpositions need to be considered only for rules having many monomials on their left-hand-sides.

3.4 GROBNER BASIS

The above process of generating critical pairs is repeated until the contradiction 1 — 0 is generated or it is no longer possible to generate any new rule. In the first case, we are done; in the second case, the original formula is falsified (respectively, satisfiable) if it is being proved valid (respectively, unsatisfiable). The Grbbner basis thus obtained provides a way to construct a model for falsifiability (satisfiability) of the formula as an example in the next section illustrates. As stated earlier, the process of generating critical pairs could continue forever for formulae which are not valid (respectively, unsatisfiable).

4. EXAMPLES

Consider a simple formula: All unicorns are quadrupeds and there is a quadruped imply that there is a unicorn. If *V* stands for something being a unicorn and *Q* stands for something being a quadruped, then this formula is:

$[(\forall x)[U(x) \Rightarrow Q(x)] \wedge (\exists x)\, Q(x)] \Rightarrow (\exists x)\, U(x).$

To show the validity of the above formula, we equate it to 0, we obtain

1. $[(\forall x)\, U(x) \Rightarrow Q(x)] = 1$ 2. $[(\exists x)\, Q(x)] = 1$
3. $[(\exists x)\, U(x)] = 0$

After Skolemization, we have:

1'. $[U(x) \Rightarrow Q(x)] = 1$ 2'. $Q(a) = 1$
3'. $U(x) = 0$

Because of $U(x) = 0$, the first equation becomes redundant. Polynomials $Q(a) \rightarrow 1$ and $U(x) \rightarrow 0$ do not generate any additional rules thus giving us a Gröbner basis. So, the original formula is not valid because we did not derive a contradiction. The original formula is falsified in a 1-element model in which $U(a) = 0$ and $Q(a) = 1$.

The following example is taken from (Chang and Lee, 1973). Consider a formula which states that (i) if there exists an element such that it is the left-identity as well as the right-identity of every element and that for every element x, $i(x)$ serves as the left-inverse as well as right-inverse of x, and (ii) if whenever x and y are in a subset P, then so is x operated with $i(y)$, then the subset P is closed under the function i.

$[(\exists y)(\forall x)[M(y,x,x) \wedge M(x,y,x) \wedge M(x,i(x),y)$
$\wedge M(i(x),x,y)] \wedge (\forall$
$x,y,z)[[P(x) \wedge P(y) \wedge M(x,i(y),z)] \Rightarrow P(z)]]$
$\Rightarrow (\forall z)[P(z) \Rightarrow P(i(z))]].$

To prove this formula to be a theorem, we assert it to 0 and deduce a contradiction. Translating the assertion to equivalent set of polynomials gives us the following rewrite rules: the constant symbols c and *b* are the Skolem functions introduced to get rid of quantifiers.

1. $M(e, x, x) \rightarrow 1$ 2. $M(x, e, x) \rightarrow 1$
3. $M(x, i(x), e) \rightarrow 1$ 4. $M(i(x), x, e) \rightarrow 1$
5. $P(b) \rightarrow 1$ 6. $P(i(b)) \rightarrow 0$
7. $P(x)\,P(y)\,P(z)\,M(x, i(y), z) \rightarrow P(x)\,P(y)\,M(x, i(y), z)$.

Variables in each rule are standardized apart. From rules 6 and 7, we get a critical pair by substituting $i(b)$ for z;

$$\begin{tikzcd}
 & P(x)\,P(y)\,P(i(b))\,M(x, i(y), i(b)) \arrow[dl] \arrow[dr] & \\
P(x)\,P(y)\,M(x, i(y), i(b)) & & 0
\end{tikzcd}$$

Let us label the new rule as 8, i.e.,

8. $P(x)\,P(y)\,M(x, i(y), i(b)) \rightarrow 0$.

Similarly, a critical pair between rules 1 and 8 using the substitution $\{ x \leftarrow e,\ y \leftarrow b \}$, after normalization, is,

9. $P(e) \rightarrow 0$.

Taking a critical pair between rules 3 and 7 using the substitution $\{ z \leftarrow e,\ x \leftarrow y \}$, gives, after normalization,

10. $P(y) \rightarrow 0$.

Using rules 5 and 10, we get the contradiction which implies that the original formula we started with is valid.

6. RELATING RESOLUTION TO THE GROBNER BASIS APPROACH

The process of generating critical pairs of first-order polynomials is similar to resolving the corresponding formulae. Below, we show that resolution can be simulated by the critical-pair-generation process. In fact, for certain formulae for which resolution does not give any meaningful result, computing critical pairs of the corresponding polynomials may still produce useful inferences.

Theorem 5.1: Let $s_1 = C_1 \vee f_1(C_2, \ldots, C_m)$, and $t_1 = \neg C_1' \vee f_2(D_2, \ldots, D_n)$ be two clauses (disjunction of literals), where C_i, $1 \leq i \leq m$, C_1', and D_j, $2 \leq j \leq n$, are positive literals. Let their binary resolvent on C_1 be $u_1 = \sigma(f_1(C_2,\ldots,C_m)) \vee \sigma(f_2(D_2,\ldots,D_n))$, where σ is the most general unifier of C_1 and C_1'. Then a polynomial form of $u_1 + 1$ can be obtained from an S-polynomial of the polynomial forms of $s_1 + 1$ and $t_1 + 1$.

Theorem 5.2: Let s be a clause and σ be a most general unifier of two literals in s of the same sign. Then a polynomial form of (*) + l can be obtained from an S-polynomial of the polynomial form of s + l and the rule (B_2)

Theorem 5.2 shows that the process of computing *factors* of clauses (cf. Chang and Lee, 1973, p. 80), which is part of the resolution method, can be simulated by repeated overlapping of polynomial rules with the rule (B_2). Thus resolution is built into the critical pair generation process as each of the four cases in the definition of a resolvent in (cf. Chang and Lee, 1973, pp. 80-81), can be simulated.

1. REFERENCES

|1] Bledsoe, W. W.,f and Tyson, M. The UT Interactive Prover," Automatic Theorem Proving Project, ATP-17A, Department of Mathematics and Computer Sciences, University of Texas, Austin, June, 1978.

|2] Buchberger, B., and Loos, R. "Algebraic Simplification" In *Computer Algebra: Symbolic and Algebraic Computation* (Eds. B. Buchberger, G.E. Collins and R. Loos), Computing Suppl. 4, Springer Verlag, New York, 1982, pp. 11-43.

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

|4] Dershowitz, N. "Orderings for Term Rewriting Systems." *Theoretical Computer Science* 17 (1982), pp. 279-301.

[5] Hsiang, J. and Dershowiti:, N. "Rewrite Methods for Clausal and Non-clausal Theorem Proving" In *Proc. 10th EATCS Intl. Colloq. on Automata, Languages, and Programming,* Spain, 1983.

[6] Kandri-Rody, A., and Kapur, D. "Computing the Grobner Basis of a Polynomial Ideal over Integers" In *Proc. Third MACSYMA (Jeers' Conference,* Schenectady, NY, July 1984, pp. 436-461.

[7] Kapur, D., and Narendran, P. "An Equational Approach to Theorem Proving in First-Order Predicate Calculus," 84CRD296, General Electric Corporate Research and Development Report, Schenectady, NY, March, 1984; Revised, Dec, 1984.

|8] Knuth, D.E. and Bendix, P.B "Simple Word Problems in Universal Algebras" In *Computational Problems in Abstract Algebras.* (Ed. J. Leech), Pergamon Press, 1970, pp 263-297

[9] Lank ford, D.S., and Ballantyne, A.M., "Decision Procedures for Simple Equational Theories with Commutative-Associative Axioms: Complete Sets of Commutative-Associative Reductions," Automatic Theorem Proving Project, Dept. of Math, and Computer Science, University of Texas, Austin, TX 78712, Report ATP-39, August 1977.

[10] Peterson, G.L., and Stickel, M.E., "Complete Sets of Reductions for Some Equational Theories," *JACM* 28 (1981), pp. 233-264.

[11] Robinson, J. A. "A Machine-Oriented Logic Based on the Resolution Principle." *JACM* 12 (1965), pp. 23-41

[12] Slagle, J. R. "Automated Theorem Proving for Theories with Simplifies, Commutativity and Associativity." *JACM* 21 (1974), pp. 622-642.

[13] van der Waerden, B.L., *Modern Algebra.* Vols. I and II, Fredrick Ungar Publishing Co., New York, 1966.