Theorem Proving Using Equational Matings and Rigid *E*-Unification

JEAN GALLIER

University of Pennsylvania, Philadelphia, Pennsylvania

PALIATH NARENDRAN

State University of New York at Albany, Albany, New York

STAN RAATZ

Rutgers University, New Brunswick, New Jersey

AND

WAYNE SNYDER

Boston University, Boston, Massachusetts

Abstract. In this paper, it is shown that the method of matings due to Andrews and Bibel can be extended to (first-order) languages with equality. A decidable version of E-unification called *rigid E-unification* is introduced, and it is shown that the method of equational matings remains complete when used in conjunction with rigid E-unification. Checking that a family of mated sets is an equational mating is equivalent to the following restricted kind of E-unification.

Problem Given $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ a family of n finite sets of equations and $S = \{\langle u_i, v_i \rangle \mid 1 \leq i \leq n\}$ a set of n pairs of terms, is there a substitution θ such that, treating each set $\theta(E_i)$ as a set of *ground* equations (i.e., holding the variables in $\theta(E_i)$ "rigid"), $\theta(u_i)$, and $\theta(v_i)$ are provably equal from $\theta(E_i)$ for $i = 1, \ldots, n$?

Equivalently, is there a substitution θ such that $\theta(u_i)$ and $\theta(v_i)$ can be shown congruent from $\theta(E_i)$ by the congruence closure method for $i = 1, \ldots, n$?

A substitution θ solving the above problem is called a *rigid $\vec{E}$-unifier of S*, and a pair $\langle \vec{E}, S \rangle$ such that S has some rigid $\vec{E}$-unifier is called an *equational premating*. It is shown that deciding whether a pair $\langle \vec{E}, S \rangle$ is an equational premating is an NP-complete problem.

This research was partially supported by the National Science Foundation under grant DCR 86-07156, and by Office of Naval Research (ONR) under grant N00014-88-K-0593.

Authors' addresses: J. Gallier, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104; P. Narendran, Department of Computer Science, State University of New York at Albany, Albany, NY 12222; S. Raatz, Department of Computer Science, Rutgers University, New Brunswick, NJ 08901; W. Snyder, Boston University, 111 Cummington Street, Boston, MA 02215.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*complexity of proof procedures*; F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*computational logic*; *mechanical theorem proving*; *proof theory*; I.1.2 [**Algebraic Manipulation**]: Algorithms—*algebraic algorithms*; I.1.3 [**Algebraic Manipulation**]: Languages and Systems—*special purpose algebraic systems*; I.2.3 [**Artificial Intelligence**]: Deduction and Theorem Proving—*deduction*; *metatheory*

General Terms: Algorithms, Languages, Theory

Additional Key Words and Phrases: Automated theorem proving, equational reasoning, Knuth–Bendix procedure, NP-completeness, matings, unification

1. *Introduction*

In this paper we show that the method of matings due to Andrews [1] and Bibel [7–10] can be extended to (first-order) languages with equality, and prove that this extension is both sound and complete.[1] A decidable version of E-unification called *rigid E-unification* is introduced, and it is shown that the method of equational matings remains complete when used in conjunction with rigid E-unification. The results of this paper extend significantly those presented at *LICS'87* [18]. In [18], it is conjectured that rigid E-unification is decidable. Subsequently, we have shown that rigid E-unification is NP-complete (*LICS'88* [20]), thus proving our conjecture. The main focus of this paper is the method of equational matings, and we present a simplified version of the decidability of rigid E-unification. Full details on the NP-completeness of rigid E-unification can be found in [22].

At first glance, a generalization of the method of matings to first-order languages with equality where equality is built-in in the sense of Plotkin [39] (thus, it is not the naive method where explicit equality axioms are added which is rejected for well-known inefficiency reasons) requires general E-unification (Gallier and Snyder [21]). Hence, there are two factors contributing to the undecidability of the method of matings for first-order languages with equality: (1) the fact that one cannot predict how many disjuncts will occur in a Herbrand expansion (which also holds for first-order languages without equality); (2) the undecidability of the kind of unification required (E-unification).

In this paper, we show that the completeness of the method of equational matings is preserved if unrestricted E-unification is replaced by rigid E-unification. We also prove that rigid E-unification is decidable, which shows that the second undecidability factor can be eliminated. The NP-completeness of rigid E-unification shows clearly how the presence of equality influences the complexity of theorem-proving methods. For languages without equality, one can use standard unification whose time complexity is polynomial, and in fact $O(n)$. For languages with equality, the unification required is NP-complete. When dealing with a fixed equational theory for which a practically tractable or decidable unification algorithm is known, we recognize that it is unclear whether our new method compares favorably with the method of matings using this specialized unification algorithm. It seems unlikely that this question can be settled at the theoretical level, and since our method has not yet been implemented, we are unable to make any claims of practicality. Nevertheless, it

[1] One of the referees has pointed out that Bibel's connection method appeared in print earlier than Andrews's method of matings.

seems unquestionable that having a decidable unification procedure (preserving completeness) represents significant progress.

The method of matings applies to formulas in negation normal form, and was introduced with two motivations in mind: to avoid breaking a formula into parts, which can result in loss of information about the global structure, and to avoid transforming it to clausal form, which can result in an exponential increase in the number of literals due to the repeated use of the distributive law $(P \vee (Q \wedge R)) \equiv ((P \vee Q) \wedge (P \vee R))$. This method is an incremental proof (or refutation) procedure that interleaves two steps: quantifier-duplication steps, and search for matings. It is an *analytic* proof procedure in the sense of Smullyan [40], and, even though Andrews did not present it in terms of Gentzen or Tableaux systems [17, 40, 41], it can easily be presented in any of these formalisms. In fact, this is the approach followed in Bibel and Schreiber [10], and thoroughly investigated in Eder [14, 15]. Fitting's method of tableaux [16] is also close in spirit to matings.

The method of matings has been implemented at CMU in the system TPS designed by Andrews and his collaborators [3]. A large number of nontrivial theorems have been proved by the system TPS, and this system is also used as an effective teaching tool. Since TPS uses a version of Huet's higher-order unification procedure [25, 26], it is capable of performing higher-order reasoning. For example, the TPS system [3] can prove Cantor's Theorem (that there is no surjection from a set to its powerset) without any assistance (the higher-order unification procedure finds a term that corresponds to the diagonal set $\{a \in A \mid a \notin f(a)\}$ used in the standard proof). Equality reasoning can be dealt with indirectly by defining equality using second-order quantifiers (see Section 5.4), but this is very inefficient, and there are no other facilities in TPS to deal directly with equality.

The method of matings exploits the fundamental property given by the Skolem–Herbrand–Gödel theorem [1, 2, 17]. In short, the unsatisfiability of a (universally) quantified sentence can be reduced to the unsatisfiability of a *quantifier-free formula*, modulo guessing a ground substitution. The crucial observation due to Andrews and Bibel is that a quantifier-free formula (in nnf) is unsatisfiable iff certain sets of literals occurring in A (called *vertical paths*) are unsatisfiable. Matings come up as a convenient method for checking that vertical paths are unsatisfiable. Roughly speaking, a *mating* is a set of pairs of literals of opposite signs (mated pairs) such that all these (unsigned) pairs are globally unified by some substitution. The importance of matings stems from the fact that a quantifier-free formula A has a mating iff there is a ground substitution θ such that $\theta(A)$ is unsatisfiable.

The extension to *equational matings* is nontrivial, and requires proving a generalization of Andrews's version of the Skolem–Herbrand–Gödel theorem [1, 2]. It also requires extending the concept of a mating so that an equational mating is a set of sets of literals (mated sets), where a mated set consists of several positive equations and a single negated equation (rather than pairs of literals, as in Andrews and Bibel's case), and a form of unification modulo equational theories (E-unification) first studied by Plotkin [39]. A related extension is sketched (without proofs) in Bibel [9, Sect. V.3, pp. 234–242]. However, Bibel's method and ours differ significantly. This is because standard unification is used in Bibel's method, and so, it is usually necessary to include extra literals arising from instances of the equality axioms to the mated sets.

On the other hand, our method uses a form of E-unification, and we *never* include any extra literals (arising from equality axioms) in our mated sets. For a detailed comparison of our method with others, see Section 5.4.

Checking that a family of mated sets is unsatisfiable, that is, an equational mating, leads to an interesting and nontrivial problem. This problem, which is central to this method, is a restricted version of E-unification.

Problem 1. Given $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ a family of n finite sets of equations and $S = \{\langle u_i, v_i \rangle \mid 1 \leq i \leq n\}$ a set of n pairs of terms, is there a substitution θ such that, treating each set $\theta(E_i)$ as a set of *ground* equations (i.e., holding the variables in $\theta(E_i)$ "rigid"), $\theta(u_i)$ and $\theta(v_i)$ are provably equal from $\theta(E_i)$ for $i = 1, \ldots, n$?

Equivalently, is there a substitution θ such that $\theta(u_i)$ and $\theta(v_i)$ can be shown congruent from $\theta(E_i)$ by the congruence closure method for $i = 1, \ldots, n$ (Kozen [30, 31], Nelson and Oppen [36], Downey et al. [13])?

A substitution θ solving problem 1 is called a *rigid $\vec{E}$-unifier* of S, and a pair $\langle \vec{E}, S \rangle$ such that S has some rigid $\vec{E}$-unifier is called an *equational premating*. It will be shown in Section 12 that deciding whether a pair $\langle \vec{E}, S \rangle$ is an equational premating is an NP-complete problem. Since the problem of deciding whether a family of mated sets forms an equational mating is equivalent to the problem of finding whether a pair $\langle \vec{E}, S \rangle$ is an equational premating, the former problem is also NP-complete. Actually, this result is an easy extension of a simpler problem.

Problem 2. Given a finite set $E = \{u_1 \doteq v_1, \ldots, u_n \doteq v_n\}$ of equations and a pair $\langle u, v \rangle$ of terms, is there a substitution θ such that, treating $\theta(E)$ as a set of ground equations, $\theta(u) \simeq_{\theta(E)} \theta(v)$, that is, $\theta(u)$ and $\theta(v)$ are congruent modulo $\theta(E)$ (by congruence closure)?

The substitution θ is called a *rigid E-unifier of u and v*.

Example 1.1. Let $E = \{fa \doteq a, ggx \doteq fa\}$, and $\langle u, v \rangle = \langle gggx, x \rangle$. Then, the substitution $\theta = [ga/x]$ is a rigid E-unifier of u and v. Indeed, $\theta(E) = \{fa \doteq a, ggga \doteq fa\}$, and $\theta(gggx)$ and $\theta(x)$ are congruent modulo $\theta(E)$, since

$$\begin{aligned} \theta(gggx) = ggggga &\rightarrow gfa \quad \text{using} \quad ggga \doteq fa \\ &\rightarrow ga = \theta(x) \quad \text{using} \quad fa \doteq a. \end{aligned}$$

Note that θ is not the only rigid E-unifier of u and v. For example, $[gfa/x]$ or more generally $[gf^k a/x]$ is a rigid E-unifier of u and v. However, θ is more general than all of these rigid E-unifiers (in a sense to be made precise later). It will be shown in Section 10 that there is always a finite set of most general rigid E-unifiers called a complete set of rigid E-unifiers.

Note that any substitution θ satisfying the above problem is an E-unifier of u and v. However, the equations in E are used in a restricted fashion. Contrary to E-unification, in which there is no bound on the number of instances of the equations in E used to show that $\theta(u) \leftrightarrow_E \theta(v)$, in our situation, *only* the m instances in $\theta(E)$ can be used (any number of times, $m \leq n$).

The solution to problem (2) is a significant extension of a result of Kozen, who has shown that the problem is NP-complete when all equations in E are ground [31]. We also show that even when u, v are ground, and all equations in E except one regular equation are ground, the problem is NP-complete.

Rigid E-unification is exciting because it eliminates one of the two aspects of undecidability associated with the method of equational matings; namely, that of E-unification. This is particularly important here, since even if E-unification is decidable for the set of all equations occurring in a formula in nnf, it is necessary to consider subsets of this set of equations, and the E-unification problem for any subset can be undecidable.

The paper is organized as follows: Section 2 reviews the main concepts used in this paper. In Section 3, the method of equational matings is presented informally by means of examples. In Section 4, the central concept of an equational mating is introduced, and some important results about them are established. Section 5 is devoted to a version of the Skolem–Herbrand–Gödel theorem for first-order languages with equality (Theorem 5.2). In order to state this theorem, we need the notion of a compound instance (see Andrews [1] and Bibel [7–9]). The connection with equational matings is made via the notion of amplification, and the completeness of the method is shown (Theorem 5.5). It is also shown that the method remains complete if outermost amplifications are performed, and the section ends with a comparison with other methods. Sections 6–12 are devoted to rigid E-unification. Basic definitions about complete sets of rigid E-unifiers are given in Section 6. Minimal rigid E-unifiers are studied in Section 7. A method for reducing a set of ground rewrite rules is reviewed in Section 8. The method for finding complete sets of rigid E-unifiers is given in Section 9. The soundness, completeness and decidability of the method are shown in Section 10. In Section 11, it is shown that rigid E-unification is NP-complete. The application of rigid E-unification to equational matings is presented in Section 12. A refutation procedure based on equational matings is presented in Section 13. Section 14 contains the conclusion. The appendix provides a semantic proof of the Skolem–Herbrand–Gödel theorem, in the line of Andrews's proof for the case without equality.

Readers who want to find out quickly about the main results (provided some familiarity with the matings/connections method) are advised to skim Section 3, then jump to Section 6, then to Section 8, Section 9, Section 12, and finally Section 13. Example 9.4 offers a simple illustration of the new method.

2. *Preliminaries*

This section contains a brief review of the main concepts used in this paper. As much as possible, we stick to the definitions used in the literature on the subject. More specifically, we follow Huet and Oppen [28], and Gallier [17]. The purpose of this section is mainly to establish the terminology and the notation, and it can be omitted by readers familiar with the literature. First, we review the basics of many-sorted languages.

Definition 2.1. A set S of *sorts* (or *types*) is any nonempty set. Typically, S consists of types in a programming language (such as *integer*, *real*, *boolean*, *character*, etc.). An *S-ranked alphabet* is a pair (Σ, ρ) consisting of a set Σ together with a function $\rho\colon \Sigma \to S^* \times S$ assigning a *rank* (u, s) to each symbol f in Σ. The string u in S^* is the *arity* of f and s is the *sort* (or *type*) of f. If $u = s_1 \cdots s_n$, $(n \geq 1)$, a symbol f of rank (u, s) is to be interpreted as an operation taking arguments, the ith argument being of type s_i and yielding a

result of type s. A symbol of rank (e, s) (when u is the empty string) is called a *constant of sort* s. For simplicity, a ranked alphabet (Σ, ρ) is often denoted by Σ.

Next, we review the definition of tree domains and trees (or terms). Let $\mathbf{N}$ denote the set of natural numbers, and $\mathbf{N}_+$ the set of positive natural numbers.

Definition 2.2. A *tree domain* D is a nonempty subset of strings in $\mathbf{N}_+^*$ satisfying the conditions:

(1) For all $u, v \in \mathbf{N}_+^*$, if $uv \in D$, then $u \in D$.
(2) For all $u \in \mathbf{N}_+^*$, for every $i \in \mathbf{N}_+$, if $ui \in D$ then, for every j, $1 \leq j \leq i$, $uj \in D$. For every $n \in \mathbf{N}$, let $[n] = \{1, 2, \ldots, n\}$, and $[0] = \varnothing$.

Definition 2.3. Given an S-sorted ranked alphabet Σ, a *Σ-tree (or term) of sort s* is any function $t\colon D \to \Sigma$, where D is a tree domain denoted by $dom(t)$, and t satisfies the following conditions:

(1) The root of t is labeled with a symbol $t(e)$ in Σ of sort s.
(2) For every node $u \in dom(t)$, if $\{i \mid ui \in dom(t)\} = [n]$, then if $n > 0$, for each ui, $i \in [n]$, if $t(ui)$ is a symbol of sort v_i, then $t(u)$ has rank (v, s'), with $v = v_1 \cdots v_n$, else if $n = 0$, then $t(u)$ has rank (e, s'), for some $s' \in S$.

Given a tree t and some tree address $u \in dom(t)$, the *subtree of t rooted at u* is the tree t/u whose domain is the set $\{v \mid uv \in dom(t)\}$ and such that $t/u(v) = t(uv)$ for all v in $dom(t/u)$.

Given two tree addresses $\alpha, \beta \in dom(t)$ in a tree t, α is an *ancestor* of β iff α is a prefix of β,[2] and α is a *proper ancestor* of β iff it is an ancestor of β and $\alpha \neq \beta$. Addresses α and β are *independent* iff neither one is an ancestor of the other. The set of all finite trees of sort s is denoted by T_Σ^s, and the S-indexed family $(T_\Sigma^s)_{s \in S}$ of all finite trees by T_Σ.

In this paper, it is assumed that for every S-sorted alphabet Σ, there is a distinguished sort $bool \in S$. Symbols of sort *bool* are called *predicate symbols*. Terms of sort *bool* will be interpreted as logical formulas.

The operation of tree replacement (or tree substitution) will be needed.

Definition 2.4. Given two trees t_1 and t_2 and a tree address u in t_1, the *result of replacing t_2 at u in t_1*, denoted by $t_1[u \leftarrow t_2]$, is the function whose graph is the set of pairs

$$\{(v, t_1(v)) \mid v \in dom(t_1),\ u \text{ is not a prefix of } v\}$$
$$\cup \{(uv, t_2(v)) \mid v \in dom(t_2)\},$$

and it is only defined provided that the sort of the root of t_2 is equal to the sort of $t_1(u)$.

Let $X = (X_s)_{s \in S}$ be an S-indexed family of countable sets of variables. We can form the S-indexed family $T_\Sigma(X)$ obtained by adjoining the S-indexed family $(X_s)_{s \in S}$ to the S-indexed family of sets of constants in Σ. To prevent free algebras from having empty carriers (so that the Herbrand–Skolem–Gödel theorem holds), we assume that every sort is *nonvoid*. We say that a sort s is *nonvoid* iff either there is some constant of sort s, or there is some function symbol f of rank $\rho(f) = (s_1, \ldots, s_n, s)$ such that $s_1, \ldots, s_n$ are nonvoid. Then,

[2] That is, $\beta = \alpha\gamma$, for some $\gamma \in \mathbf{N}_+^*$.

for every sort s, the set T_Σ^s is nonempty, and it is well known that for every set X, $T_\Sigma(X)$ is the free Σ-algebra generated by X (see Gallier [17]). This allows us to define substitutions.

Definition 2.5. Given a term t, the set of variables occurring in t is the set $\{x \in X \mid \exists u \in dom(t),\ t(u) = x\}$, and it is denoted by $Var(t)$.

Definition 2.6. A *substitution* is any (S-sorted) function $\sigma\colon X \to T_\Sigma(X)$, such that, $\sigma(x) \neq x$ for only finitely many $x \in X$. Since $T_\Sigma(X)$ is the free Σ-algebra generated by X, every substitution $\sigma\colon X \to T_\Sigma(X)$ has a unique homomorphic extension $\hat{\sigma}\colon T_\Sigma(X) \to T_\Sigma(X)$. In the sequel, we identify σ and its homomorphic extension $\hat{\sigma}$.

Definition 2.7. Given a substitution σ, the *support* (or *domain*) of σ is the set of variables $D(\sigma) = \{x \mid \sigma(x) \neq x\}$. The set of variables *introduced by* σ is the set of variables $I(\sigma) = \bigcup_{x \in D(\sigma)} Var(\sigma(x))$. Given a substitution σ, if its support is the set $\{x_1, \ldots, x_n\}$, and if $t_i = \sigma(x_i)$, $1 \leq i \leq n$, then σ is also denoted by $[t_1/x_1, \ldots, t_n/x_n]$. Given a term (or formula) r, we also denote $\sigma(r)$ as $r[t_1/x_1, \ldots, t_n/x_n]$.

Given a substitution σ and a set W of variables, the *restriction* of σ to W, denoted by $\sigma \mid_W$, is the substitution θ defined such that, $\theta(x) = \sigma(x)$ for all $x \in W$, and $\theta(x) = x$ for all $x \notin W$.

Definition 2.8. Given two substitutions σ and θ, their *composition* is the substitution denoted by $\sigma;\ \theta$, such that, for every variable x, $\sigma;\ \theta(x) = \hat{\theta}(\sigma(x))$ (the composition of the functions σ and $\hat{\theta}$).

A substitution σ is *idempotent* if $\sigma;\ \sigma = \sigma$. It is easily seen that σ is idempotent iff $D(\sigma) \cap I(\sigma) = \emptyset$.

We also quickly review formulas in negation normal form. For details, see Gallier [17].

Definition 2.9. An atomic formula is a term of the form either $Pt_1 \cdots t_n$, where P is a predicate symbol of rank $(s_1, \ldots, s_n, bool)$ and each t_i is a term of sort s_i ($s_i \neq bool$), or a term of the form $(t_1 \doteq t_2)$, where t_1 and t_2 are terms of some identical sort s ($s \neq bool$). An atomic formula of the form $(t_1 \doteq t_2)$ is called an *equation of sort s*. It is assumed that *bool* never occurs in the arity of any symbol. A *literal* is either an atomic formula or the negation of an atomic formula.

Definition 2.10. Formulas in *negation normal form* (for short, formulas in *nnf*) are defined inductively as follows: A formula A is in nnf iff either

(1) A is a literal, or
(2) $A = (B \vee C)$, where B and C are in nnf, or
(3) $A = (B \wedge C)$, where B and C are in nnf, or
(4) $A = \forall xB$, where B is in nnf, or
(5) $A = \exists xB$, where B is in nnf.

A *quantifier-free* formula in nnf is obtained by applying only clauses (1)–(3), and a *universal* formula in nnf by applying only clauses (1)–(4).

Definition 2.11. Given a formula A (resp., a term t), the set of variables occurring free in A (resp., t) is denoted by $Var(A)$ (resp., $Var(t)$). A *ground term* t is a term such that $Var(t) = \emptyset$, and similarly a *ground formula* A is a

quantifier-free formula such that $Var(A) = \emptyset$. A *ground substitution* σ is a substitution such that $\sigma(x)$ is a ground term for every variable x in the support of σ.

Finally, we review some concepts related to term rewriting.

Definition 2.12. Let $\rightarrow$ be a binary relation $\rightarrow \subseteq A \times A$ on a set A. The transitive closure of $\rightarrow$ is denoted by $\overset{+}{\rightarrow}$ and the reflexive and transitive closure of $\rightarrow$ by $\overset{*}{\rightarrow}$. The *converse* (or *inverse*) of the relation $\rightarrow$ is the relation denoted as $\rightarrow^{-1}$ or $\leftarrow$, defined such that $u \leftarrow v$ iff $v \rightarrow u$. The symmetric closure of $\rightarrow$, denoted by $\leftrightarrow$, is the relation $\rightarrow \cup \leftarrow$.

Definition 2.13. A relation $\succ$ on a set A is *Noetherian* or *well founded* iff there are no infinite sequences $\langle a_0, \ldots, a_n, a_{n+1}, \ldots \rangle$ of elements in A such that $a_n \succ a_{n+1}$ for all $n \geq 0$.[3]

Definition 2.14. A *preorder* $\preceq$ on a set A is a binary relation $\preceq \subseteq A \times A$ that is reflexive and transitive. A *partial order* $\preceq$ on a set A is a preorder that is also antisymmetric. The converse of a preorder (or partial order) $\preceq$ is denoted as $\succeq$. A *strict ordering* (or *strict order*) $\prec$ on a set A is a transitive and irreflexive relation. Given a preorder (or partial order) $\preceq$ on a set A, the strict ordering $\prec$ associated with $\preceq$ is defined such that $s \prec t$ iff $s \preceq t$ and $t \not\preceq s$. Conversely, given a strict ordering $\prec$, the partial ordering $\preceq$ associated with $\prec$ is defined such that $s \preceq t$ iff $s \prec t$ or $s = t$. The converse of a strict ordering $\prec$ is denoted as $\succ$. Given a preorder (or partial order) $\preceq$, we say that $\preceq$ is well founded iff $\succ$ is well founded.[4]

Definition 2.15. Let $\rightarrow$ be a binary relation $\rightarrow \subseteq T_\Sigma(X) \times T_\Sigma(X)$ on terms. The relation $\rightarrow$ is *monotonic* iff for every two terms s, t and every function symbol f, if $s \rightarrow t$ then $f(\ldots, s, \ldots) \rightarrow f(\ldots, t, \ldots)$. The relation $\rightarrow$ is *stable* (under substitution) if $s \rightarrow t$ implies $\sigma(s) \rightarrow \sigma(t)$ for every substitution σ.

Definition 2.16. A strict ordering $\prec$ has the *subterm property* iff $s \prec f(\ldots, s, \ldots)$ for every term $f(\ldots, s, \ldots)$ (since we are considering symbols having a fixed rank, the deletion property is superfluous, as noted in Dershowitz [11]). A *simplification ordering* $\prec$ is a strict ordering that is monotonic and has the subterm property. A *reduction ordering* $\prec$ is a strict ordering that is monotonic, stable, and such that $\succ$ is well founded. With a slight abuse of language, we also say that the converse $\succ$ of a strict ordering $\prec$ is a simplification ordering (or a reduction ordering). It is shown in Dershowitz [11] that there are simplification orderings that are total on ground terms.

Definition 2.17. Let $E \subseteq T_\Sigma(X) \times T_\Sigma(X)$ be a binary relation on terms. We define the relation $\leftrightarrow_E$ over $T_\Sigma(X)$ as the smallest symmetric, stable, and

[3] We warn the readers that this is not the usual way of defining a well-founded relation in set theory, as for example in Levy [32]. In set theory, the condition is stated in the form $a_{n+1} \prec a_n$ for all $n \geq 0$, where $\prec = \succ^{-1}$. It is the dual of the condition we have used, but since $\prec = \succ^{-1}$, the two definitions are equivalent. When using well-founded relations in the context of rewriting systems, we are usually interested in the reduction relation $\Rightarrow$ and the fact that there are no infinite sequences $\langle a_0, \ldots, a_n, a_{n+1}, \ldots \rangle$ such that $a_n \Rightarrow a_{n+1}$ for all $n \geq 0$. Thus, following other authors, including Dershowitz, we adopt the dual of the standard set theoretic definition.

[4] Again, we caution our readers that in standard set theory it is $\prec$ that is well founded! However, our definition is equivalent to the standard set-theoretic definition of a well-founded partial ordering.

monotonic relation that contains E. This relation is defined explicitly as follows: Given any two terms $t_1, t_2 \in T_\Sigma(X)$, then $t_1 \leftrightarrow_E t_2$ iff there is some variant[5] (s, t) of a pair in $E \cup E^{-1}$, some tree address α in t_1, and some substitution σ, such that

$$t_1/\alpha = \sigma(s) \qquad \text{and} \qquad t_2 = t_1[\alpha \leftarrow \sigma(t)].$$

(In this case, we say that σ is a *matching substitution* of s onto t_1/α. The term t_1/α is called a *redex*.) Note that the pair (s, t) is used as a two-way rewrite rule (that is, nonoriented). In such a case, we denote the pair (s, t) as $s \doteq t$ and call it an *equation*. When $t_1 \leftrightarrow_E t_2$, we say that we have an *equality step*. It is well known that the reflexive and transitive closure $\overset{*}{\leftrightarrow}_E$ of $\leftrightarrow_E$ is the smallest stable congruence on $T_\Sigma(X)$ containing E. When we want to fully specify an equality step, we use the notation

$$t_1 \leftrightarrow_{[\alpha, s=t, \sigma]} t_2$$

(where some of the arguments may be omitted). A sequence of equality steps

$$u = u_0 \leftrightarrow_E u_1 \leftrightarrow_E \cdots \leftrightarrow_E u_{n-1} \leftrightarrow_E u_n = v$$

is called a *proof* of $u \overset{*}{\leftrightarrow}_E v$.

Definition 2.18. Given a finite set E of equations (ground or not) and any two terms u, v (ground or not), we use the notation $u \overset{*}{=}_E v$ to express the fact that, treating E as a set of *ground equations*, $u \overset{*}{\leftrightarrow}_E v$. Equivalently, $u \overset{*}{=}_E v$ iff u and v can be shown congruent from E by *congruence closure* (Kozen [30, 31], Nelson and Oppen [36], Downey et al. [13]) again, treating all variables as constants—they are considered *rigid*.

Definition 2.19. When a pair $(s, t) \in E$ is used as an oriented equation (from left to right), we call it a *rule* and denote it as $s \to t$. The *reduction relation* $\to_E$ is the smallest stable and monotonic relation that contains E. We can define $t_1 \to_E t_2$ explicitly as in Definition 2.17, the only difference being that (s, t) is a variant of a pair in E (and not in $E \cup E^{-1}$). When $t_1 \to_E t_2$, we say that t_1 *rewrites* to t_2, or that we have a *rewrite step*. When we want to fully specify a rewrite step, we use the notation

$$t_1 \to_{[\alpha, s \to t, \sigma]} t_2$$

(where some of the arguments may be omitted).

When $Var(r) \subseteq Var(l)$, then a rule $l \to r$ is called a *rewrite rule*; a set of such rules is called a *rewrite system*. A *degenerate equation* is an equation of the form $x \doteq t$, where x is a variable and $x \notin Var(t)$, and a *nondegenerate equation* is an equation that is not degenerate.

Definition 2.20. Let $\to \subseteq T_\Sigma(X) \times T_\Sigma(X)$ be a binary relation on $T_\Sigma(X)$. We say that $\to$ is *Church–Rosser* iff for all $t_1, t_2 \in T_\Sigma(X)$, if $t_1 \overset{*}{\leftrightarrow} t_2$, then there is some $t_3 \in T_\Sigma(X)$ such that $t_1 \overset{*}{\to} t_3$ and $t_2 \overset{*}{\to} t_3$. We say that $\to$ is *confluent* iff for all $t, t_1, t_2 \in T_\Sigma(X)$, if $t \overset{*}{\to} t_1$ and $t \overset{*}{\to} t_2$, then

[5] In what follows, we shall assume that before a pair (i.e., an equation) is used it has been renamed apart from all variables in current use. This is essential to prevent clashes among the variables. Thus, we shall always state that a *variant* of an equation is being used.

there is some $t_3 \in T_\Sigma(X)$ such that $t_1 \overset{*}{\rightarrow} t_3$ and $t_2 \overset{+}{\rightarrow} t_3$. A term s is *irreducible* with respect to $\rightarrow$ iff there is no term t such that $s \rightarrow t$.

It is well known that a relation is confluent iff it is Church–Rosser [27]. We say that a rewrite system R is Noetherian, Church–Rosser, or confluent, iff the relation $\rightarrow_R$ associated with R given in Definition 2.19 has the corresponding property. We say that R is *canonical* iff it is Noetherian and confluent.

3. *Review of the Method*

In this section, we present the method of equational matings informally. Given a rectified[6] universal sentence A_0 in nnf, the method works in an incremental fashion as follows. The formula A_0 will evolve in steps called *quantifier duplication steps*, and we denote this evolving formula by A. We also need to refer to the quantifier-free formula $\hat{A}$ obtained from A by deleting the quantifiers, called an *amplification* of A_0.

Initially, $A := A_0$.

Step 1. Construct a set $vp(\hat{A})$ of sets of literals called vertical paths, associated with $\hat{A}$. The set $vp(\hat{A})$ is defined inductively as follows:

If $\hat{A}$ is a literal, then $vp(\hat{A}) = \{\{\hat{A}\}\}$;
If $\hat{A} = (B \wedge C)$, then $vp(\hat{A}) = \{\pi_1 \cup \pi_2 \mid \pi_1 \in vp(B), \pi_2 \in vp(C)\}$;
If $\hat{A} = (B \vee C)$, then $vp(\hat{A}) = vp(B) \cup vp(C)$.

Step 2. Find whether there is a substitution σ such that for every vertical path $\pi \in vp(\hat{A})$, $\sigma(\pi)$ is unsatisfiable. If Step 2 succeeds, go to Step 4. Otherwise, go to Step 3.

Step 3. Choose some universal subformula $\forall xB$ of the current A, and replace it by $(\forall xB \wedge \forall xB)$. Then, rectify variables in this new formula, obtaining A'. Let $A := A'$. This step is called a *quantifier duplication step*. Go back to Step 1.

Step 4. Stop, A_0 is unsatisfiable (and so are $\hat{A}$ and A).

If A_0 is unsatisfiable, this procedure will stop after a finite number of quantifier duplication steps when it succeeds in finding some substitution closing all vertical paths in Step 2. Roughly speaking, a set consisting of certain subsets of vertical paths, such that these subsets are unsatisfiable under some substitution and span all vertical paths, is called an *equational mating*. The heart of the method of matings is to find such equational matings.

The difficult step in the presence of equality is Step 2. What is difficult is not to check that a substitution closes all vertical paths—this can be done using the congruence closure algorithm—but to determine whether such a substitution exists at all. This problem is indeed decidable, but NP-complete. For languages without equality, the checking is reduced to the existence of a standard unifier, which is easy. Unfortunately, whether or not

[6] A formula A is *rectified* iff no variable occurs both free and bound in A, and distinct occurrences of quantifiers bind distinct variables. It is well known that every formula is equivalent to a rectified formula. It is also well known that for every formula A, one can construct a universal formula B, a *Skolem form* of A, such that A is unsatisfiable iff B is unsatisfiable (see Gallier [17]).

equality is present, the number of vertical paths to be checked may be exponential. The following example illustrates the method.

Example 3.1. Let

$$A = \big(\forall x(x^2 \doteq x) \wedge ((Pa^3 \wedge \neg Pa) \vee (Pb^2 \wedge \neg Pb))\big).$$

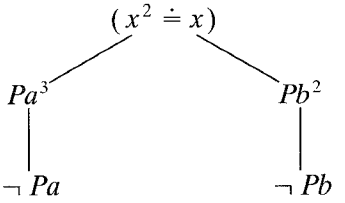
There are two vertical paths

$$\{(x^2 \doteq x), Pa^3, \neg Pa\}$$

and

$$\{(x^2 \doteq x), Pb^2, \neg Pb\}$$

depicted as follows:



It is clear that there is no substitution that closes both paths. However, the substitution $[a/x]$ closes the first path, and the substitution $[b/x]$ closes the second path. Hence, we perform an amplification step. We obtain

$$A' = \Big(\big(\forall x_1(x_1^2 \doteq x_1) \wedge \forall x_2(x_2^2 \doteq x_2)\big) \wedge ((Pa^3 \wedge \neg Pa) \vee (Pb^2 \wedge \neg Pb))\Big).$$

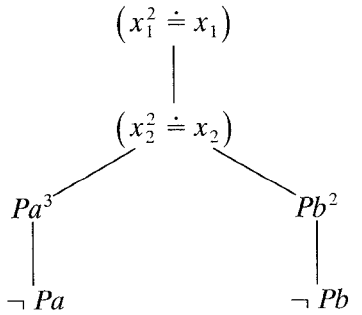
There are two vertical paths

$$\{(x_1^2 \doteq x_1), (x_2^2 \doteq x_2), Pa^3, \neg Pa\}$$

and

$$\{(x_1^2 \doteq x_1), (x_2^2 \doteq x_2), Pb^2, \neg Pb\}$$

depicted as follows:



This time, it is easy to see that the substitution $\sigma = [a/x_1, b/x_2]$ closes both vertical paths, using the fact that Pa^3 rewrites to Pa in two steps using the equation $a^2 \doteq a$, and that Pb^2 rewrites to Pb in one step using the equation $b^2 \doteq b$. Hence, A is unsatisfiable.

It should be noted that our method does not require the inclusion of extra literals corresponding to instances of equational axioms during the amplifica-

tion process, contrary to Bibel's method [9]. In this sense, equality is "built in". In the following sections, we shall define the method precisely and prove its completeness rigorously.

4. *Equational Matings*

In order to generalize matings to equational languages, it is necessary to consider sets of literals rather than pairs, as in Andrews and Bibel's case. Let us first consider the case of *quantifier-free formulas* in negation normal form. The general case will be lifted from the quantifier-free case via the Skolem–Herbrand–Gödel theorem, and using rigid E-unification.

Let A be a quantifier-free formula, and let $\{x_1, \ldots, x_n\}$ be the set of variables occurring in A. The *universal closure* of A is the sentence $\forall x_1 \cdots \forall x_n A$. It is also denoted as $\forall A$. Testing the unsatisfiability of a quantifier-free formula A is much easier than testing the unsatisfiability of its universal closure $\forall A$. In the former case, the congruence closure method gives a decision procedure, whereas in the latter case, unsatisfiability is undecidable.

The crucial observation due to Andrews and Bibel is that a quantifier-free formula in nnf is satisfiable iff some conjunction of literals occurring in A is satisfiable [1, 7, 8, 9].

Definition 4.1. Given a quantifier-free formula A in nnf, the set $vp(A)$ of *vertical paths* in A is the set of sets of literals defined inductively as follows:

If A is a literal, then $vp(A) = \{\{A\}\}$;
If $A = (B \wedge C)$, then $vp(A) = \{\pi_1 \cup \pi_2 \mid \pi_1 \in vp(B), \pi_2 \in vp(C)\}$;
If $A = (B \vee C)$, then $vp(A) = vp(B) \cup vp(C)$.

Let us say that a vertical path π is satisfiable iff the conjunction of the literals in π is satisfiable. The following simple lemma shows the crucial role played by vertical paths.

LEMMA 4.2. *Given a quantifier-free formula A in nnf, A is unsatisfiable iff every vertical path in A is unsatisfiable.*

PROOF. Straightforward induction on the structure of A. □

A criterion for the unsatisfiability of a conjunction of literals based on the concept of congruence closure is known. In order to explain this criterion, it is convenient to represent every atomic formula as an equation. This can be done by adding to our language (which already contains the special sort *bool*) the constant $\top$ of sort *bool*, interpreted as **true**. Then, every atomic formula $Pt_1, \ldots, t_n$ of sort *bool* can be expressed as the equation $(Pt_1, \ldots, t_n \doteq \top)$. Hence, we can assume that all atomic formulas are equations. The notations $Pt_1, \ldots, t_n$ and $(Pt_1, \ldots, t_n \doteq \top)$ will be used interchangeably for atomic formulas of sort *bool*.

Given a vertical path π, let us arrange the literals in π by grouping positive and negative literals together, to form a conjunction C_π of the form

$$(s_1 \doteq t_1) \wedge \cdots \wedge (s_m \doteq t_m) \wedge \neg(s'_1 \doteq t'_1) \wedge \cdots \wedge \neg(s'_n \doteq t'_n).$$

Let $\doteq_E$ be the congruence closure [17, 29, 30] on the graph $G(C_\pi)$ of the relation

$$E = \{(s_1, t_1), \ldots, (s_m, t_m)\}.$$

The following result is well known (see [17], [29], and [30]).

LEMMA 4.3. *π is unsatisfiable iff for some j, $1 \leq j \leq n$, $s'_j \doteq_E t'_j$.*

The concept of an E-unifier will be needed later.

Definition 4.4. Let E be a finite set of (universally quantified) equations. Given two terms u and v, we say that a substitution σ is a *unifier of u and v modulo E*, for short, an *E-unifier of u and v*, iff $\sigma(u) \overset{\cdot}{\leftrightarrow}_E \sigma(v)$.

The definition of an equational mating is motivated by Lemma 4.3, the Skolem–Herbrand–Gödel theorem (Theorem 5.2), and Lemma 5.4. Indeed, combining Theorem 5.2 and Lemma 5.4, we have that a universal sentence A in nnf is unsatisfiable iff there is some quantifier-free formula D (an amplification of A) and some substitution σ such that $\sigma(D)$ is unsatisfiable. The concept of an equational mating is designed so that we have a criterion expressed in terms of vertical paths for testing whether given a quantifier-free formula D, there is some substitution σ such that $\sigma(D)$ is unsatisfiable (see Lemma 4.6).

Definition 4.5. Let A be a quantifier-free formula in nnf. An *equational mating* $\mathcal{M}$ for A is a pair $\langle MS, \sigma \rangle$, where MS is a set of sets of literals called *mated sets* and σ is a substitution, such that, each mated set is a subset of some vertical path $\pi \in vp(A)$ and is of the form

$$\mathcal{S} = \{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \subseteq \pi,$$

where $m \geq 0$,[7] and, for every mated set $\{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \in MS$, the set of literals $\{\sigma(s_1 \doteq t_1), \ldots, \sigma(s_m \doteq t_m), \neg\, \sigma(s \doteq t)\}$ is unsatisfiable. The substitution associated with the mating $\mathcal{M}$ is also denoted as $\sigma_{\mathcal{M}}$. We also commit a slight abuse of language (and notation) and say that a mated set belongs to $\mathcal{M}$.

An equational mating $\mathcal{M}$ is a *refutation mating* iff $\sigma_{\mathcal{M}}(A)$ is unsatisfiable.

An equational mating $\mathcal{M}$ is *path acceptable*[8] (for short, *p-acceptable*), iff, for every path $\pi \in vp(A)$, there is some mated set $\{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \in \mathcal{M}$, such that

$$\{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \subseteq \pi.$$

A number of remarks are in order:

(1) Given the substitution σ, the mating condition can be tested using the congruence closure method. As mentioned in the introduction, it is decidable whether a mating substitution exists, but this is an NP-complete problem.

(2) Given a family MS of mated sets, let $\vec{E} = (E_{\mathcal{S}})_{\mathcal{S} \in MS}$ be the family of sets of equations of the form $E_{\mathcal{S}} = \{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m)\}$ and $S = \{\langle s, t \rangle \mid \mathcal{S} \in MS\}$ the set of pairs where $E_{\mathcal{S}}$ and $\langle s, t \rangle$ are associated with the mated sets $\mathcal{S} = \{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \in MS$. Observe that $\mathcal{M} = \langle MS, \sigma \rangle$ is a mating iff σ is a solution of problem 1 (discussed in the introduction) for $\langle \vec{E}, S \rangle$, iff $\langle \vec{E}, S \rangle$ is an equational premating. This key observation will be used in searching for the substitutions associated with matings. They are the rigid $\vec{E}$-unifiers of S.

[7] The case $m = 0$ is indeed possible when $\sigma(s) = \sigma(t)$, that is, when σ is a unifier of s and t.

[8] A path acceptable mating is also called a *spanning mating* by Miller [35].

(3) It is obviously desirable to choose p-acceptable matings as small as possible. One can define a preorder on matings as follows. Given two matings $\mathscr{M}_1$ and $\mathscr{M}_2$, $\mathscr{M}_1 \sqsubseteq \mathscr{M}_2$ iff for every mated set $S_1 \in \mathscr{M}_1$, there is some mated set $S_2 \in \mathscr{M}_2$, such that, $S_1 \subseteq S_2$. A mating $\mathscr{M}$ is *minimal* iff it is minimal with respect to the preorder $\sqsubseteq$, that is, for any mating $\mathscr{M}'$, if $\mathscr{M}' \sqsubseteq \mathscr{M}$, then $\mathscr{M}' = \mathscr{M}$. It is obvious that if a p-acceptable mating exists, then a minimal p-acceptable mating also exists, but it may be difficult to find it efficiently. In order to find matings as small as possible, one can look for overlapping vertical paths that are spanned by some common mated set. It should be pointed out that there may be many incomparable matings that are all minimal. We leave the problem of discovering strategies for finding minimal matings as a topic for further research.

(4) If A does not contain equations, each mated set contains some atom that unifies with the negated atom. Let $Pt_1 \cdots t_n$ be the negated atom in a mated set. Any mated set for a formula without equational atoms is of the form $\{(A_1 \doteq \top), \ldots, (A_m \doteq \top), \neg(Pt_1 \cdots t_n \doteq \top)\}$, where $A_1, \ldots, A_m$ are nonequational atoms. Since the set $\{\sigma(A_1 \doteq \top), \ldots, \sigma(A_m \doteq \top), \neg\sigma(Pt_1 \cdots t_n \doteq \top)\}$ is unsatisfiable, there is some atom $A_i = Ps_1 \cdots s_n$, such that, $\sigma(Pt_1 \cdots t_n) = \sigma(Ps_1 \cdots s_n)$. Hence, σ is a unifier of $Pt_1 \cdots t_n$ and $Ps_1 \cdots s_n$.

Hence, when A does not contain equality, a mating can be defined as a set of pairs $\langle L, \neg L' \rangle$ of literals of opposite signs, such that $\sigma(L) = \sigma(L')$, as in Andrews [1] and Bibel [7–9]. The following theorem is a straightforward generalization of a result of Andrews [1] to languages with equality.

LEMMA 4.6. *Given a quantifier-free formula A in nnf, the following properties hold*:

(1) *Given a substitution θ, if $\theta(A)$ is unsatisfiable, then there is a p-acceptable mating $\mathscr{M}$ for A.*

(2) *A p-acceptable mating $\mathscr{M}$ for A is a refutation mating for A, that is, $\sigma_{\mathscr{M}}(A)$ is unsatisfiable.*

PROOF

(1) Assume that $\theta(A)$ is unsatisfiable. By Lemma 4.2, every vertical path in $vp(\theta(A))$ is unsatisfiable. Note that every vertical path $\pi' \in vp(\theta(A))$ is of the form $\theta(\pi)$, for some vertical path $\pi \in vp(A)$. Since every path $\pi' \in vp(\theta(A))$ is unsatisfiable, by Lemma 4.3, there is some subset $\{(s'_1 \doteq t'_1), \ldots, (s'_m \doteq t'_m), \neg(s' \doteq t')\} \subseteq \pi'$ of literals in π' that is unsatisfiable. For every vertical path $\pi \in vp(A)$, since $\pi' = \theta(\pi)$ is a vertical path in $vp(\theta(A))$, we can choose a set of literals $\{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n), \neg(s \doteq t)\} \subseteq \pi$, such that,

$$\{\theta(s_1 \doteq t_1), \ldots, \theta(s_n \doteq t_n), \neg\theta(s \doteq t)\} = \{(s'_1 \doteq t'_1), \ldots, (s'_m \doteq t'_m), \neg(s' \doteq t')\} \qquad (*)$$

is unsatisfiable. We form a mating $\mathscr{M} = \langle MS, \theta \rangle$ for A by choosing MS as the set of sets of literals defined in $(*)$. Clearly, $\mathscr{M}$ is a p-acceptable mating for A.

(2) Assume that $\mathscr{M} = \langle MS, \sigma \rangle$ is a p-acceptable mating for A. We prove that every vertical path $\pi' \in vp(\sigma(A))$ is unsatisfiable. Indeed, every vertical path $\pi' \in vp(\sigma(A))$ is of the form $\sigma(\pi)$, for some vertical path $\pi \in vp(A)$.

Since $\mathcal{M}$ is p-acceptable, for every vertical path $\pi \in vp(A)$, there is some mated set of literals $\{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \in \mathcal{M}$, such that

$$\{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \subseteq \pi.$$

Since $\mathcal{M}$ is a mating, the set $S_{\pi'} = \{\sigma(s_1 \doteq t_1), \ldots, \sigma(s_m \doteq t_m), \neg\sigma(s \doteq t)\}$ is unsatisfiable. Since $S_{\pi'}$ is a subset of the vertical path $\pi' \in vp(\sigma(A))$, π' is unsatisfiable. But then, by Lemma 4.2, $\sigma(A)$ is unsatisfiable, which establishes the fact that $\mathcal{M}$ is a refutation mating. □

The previous lemma implies the following useful corollary.

COROLLARY 4.7. *Given a quantifier-free formula A in nnf, there is a substitution θ such that $\theta(A)$ is unsatisfiable iff there is a p-acceptable mating $\mathcal{M}$ for A.*

Let us give an example illustrating the use of the previous lemma.

Example 4.8. Consider the following Horn formula A, where x, y, z denote variables:

$$\begin{aligned}
&(a \doteq b) \wedge \\
&\big((f^3x \doteq x) \vee \neg(fx \doteq fb)\big) \wedge \\
&\qquad \big(Qa \vee \neg(f^3a \doteq a)\big) \wedge \\
&\big((f^5y \doteq y) \vee \neg Qy\big) \wedge \\
&\qquad (Ra \vee \neg(fa \doteq a) \vee \neg Pfa) \wedge \\
&\qquad\qquad \neg Rfz \wedge \\
&\quad Pa
\end{aligned}$$

There are 24 vertical paths in A. Let $\theta = [a/x, a/y, a/z]$. The substitution θ closes all the paths in $\theta(A)$, which is easy to see for the 21 vertical paths containing the sets of literals $\{(f^3a \doteq a), \neg(f^3a \doteq a)\}$, $\{Qa, \neg Qa\}$, and $\{(a \doteq b), \neg(fa \doteq fb)\}$. A p-acceptable mating for A is given by θ and the following set of 6 sets of literals:

$$\begin{gathered}
\big\{\big\{(f^3x \doteq x), \neg(f^3a \doteq a)\big\}, \\
\{Qa, \neg Qy\}, \\
\{(a \doteq b), \neg(fx \doteq fb)\}, \\
\big\{(f^5y \doteq y), (f^3x \doteq x), Ra, \neg Rfz\big\}, \\
\big\{(f^5y \doteq y), (f^3x \doteq x), \neg(fa \doteq a)\big\}, \\
\big\{(f^5y \doteq y), (f^3x \doteq x), Pa, \neg Pfa\big\}\big\}.
\end{gathered}$$

The above set is a mating because $(fa \doteq a)$ is equationally provable from $(f^3a \doteq a)$ and $(f^5a \doteq a)$. Indeed, $(f^3a \doteq a)$ implies $(f^4a \doteq fa)$, which implies $(f^5a \doteq f^2a)$, which, by transitivity, implies $(f^2a \doteq a)$. In turn, $(f^2a \doteq a)$ implies $(f^3a \doteq fa)$, and by one more application of transitivity, this implies $(fa \doteq a)$. According to Lemma 4.6, $\theta(A)$ is unsatisfiable. Since $\models \forall x\, \forall y\, \forall z A \supset \theta(A)$, the universal closure $\forall A$ of A is also unsatisfiable.

Unfortunately, in general, a universal sentence $\forall A$ may be unsatisfiable, but there may not be any substitution θ such that $\theta(A)$ is unsatisfiable (see Example 3.1). However, a version of the Skolem–Herbrand–Gödel theorem for first-order languages with equality ensures that some substitution instance of an *amplification* of A (a formula obtained from A by duplicating some universal subformulas of A) is unsatisfiable. It is the notion of amplification (see Andrews [1] and Bibel [7–9]), that will allow us to apply the method of matings to arbitrary universal sentences in nnf.

5. *A Skolem–Herbrand–Gödel Theorem*

First, we need the definition of a compound instance (see Andrews [1] and Bibel [7–9]).

5.1 COMPOUND INSTANCES. From now on, it is assumed that we are dealing with *rectified universal formulas in nnf*. The standard statement of the Skolem–Herbrand–Gödel theorem (as in Gallier [17]) says that given a universal *prenex* sentence $A = \forall x_1 \cdots \forall x_n B$ (where B is quantifier-free), A is unsatisfiable iff there exist some ground substitutions $\sigma_1, \ldots, \sigma_k$ such that $\sigma_1(B) \wedge \cdots \wedge \sigma_k(B)$ is unsatisfiable.

It would be nice if we could relax the condition that A is in prenex form, and have a statement referring to a single substitution. This can be achieved by introducing the ingenious concepts of a compound instance and of an amplification (see Andrews [1] and Bibel [7–9]).

Definition 5.1. Let A be a rectified universal sentence in nnf (Every variable occurring in A is universally quantified). The set of *compound instances* (for short, *c-instances*) of A is defined inductively as follows:

(i) If A is either a ground atomic formula B or the negation $\neg B$ of a ground atomic formula, then A is its only c-instance;

(ii) If A is of the form $(B * C)$, where $* \in \{\vee, \wedge\}$, for any c-instance H of B and c-instance K of C, $(H * K)$ is a c-instance of A;

(iii) If A is of the form $\forall x B$, for any $k \geq 1$ ground terms $t_1, \ldots, t_k$, if H_i is a c-instance of $B[t_i/x]$ for $i = 1, \ldots, k$, then $H_1 \wedge \cdots \wedge H_k$ is a c-instance of A.

The importance of c-instances lies in the following version of the Skolem–Herbrand–Gödel theorem, which is a generalization of a theorem of Andrews to first-order languages with equality [1, 2]. For stating this theorem, we assume (without loss of generality) that there is a least one constant symbol in the language.

THEOREM 5.2 (Skolem–Herbrand–Gödel theorem). *Given a universal sentence A in nnf (with or without equality), A is unsatisfiable iff some c-instance C of A is unsatisfiable.*

PROOF. It is nontrivial. A proof is given in Gallier [17, Theorem 7.6.1, page 364]. Showing that if some compound instance C is unsatisfiable implies that A is unsatisfiable is straightforward, because it is easily shown that $\models A \supset C$ [17, Theorem 7.6.1]. The proof of the converse is much harder. In Gallier [17], this is derived proof-theoretically as a consequence of a sharpened Gentzen-like Hauptsatz [17, Theorem 7.4.1, page 334, Theorem 7.4.2, page 337, and Lemma 7.6.2, page 360]. For the sake of completeness, a semantic proof in the line of Andrews's proof can be found in the appendix. □

The connection between matings and compound instances is established through the notion of amplification (see Andrews [1] and Bibel [7–9]).

5.2 AMPLIFICATIONS AND COMPOUND INSTANCES. Let A, B, C, and D be universal sentences in nnf.

Definition 5.3. We say that a sentence C is obtained from a sentence B by *quantifier duplication* iff C results from B by replacing some subformula of B of the form $\forall xM$ by $(\forall xM \wedge \forall xM)$. If there is a sequence $\langle C_1, \ldots, C_n \rangle$, $n \geq 1$, of formulas, such that, $B = C_1$, $C = C_n$, and C_{i+1} is obtained from C_i by quantifier duplication, for every $i, 1 \leq i < n$, we say that C is obtained from B by some *sequence of quantifier duplications*.

If B is obtained from A by some sequence of quantifier duplications, C is a rectified sentence equivalent to B, and D is obtained from C by deleting the quantifiers in C, we say that D is an *amplification* of A. The following result can be shown easily.

LEMMA 5.4. *Given a universal sentence A in nnf, C is a c-instance of A iff there is some amplification D of A and some (ground) substitution θ such that $C = \theta(D)$.*

PROOF. The proof is by induction on the structure of A. The only case worth mentioning is the case in which $A = \forall xB$. In this case, there are k ground terms $t_1, \ldots, t_k$ and k formulas $H_1, \ldots, H_k$, such that, each H_i is a c-instance of $B[t_i/x]$, and $C = H_1 \wedge \cdots \wedge H_k$. By the induction hypothesis, for each i, $1 \leq i \leq k$, there is some amplification D_i of $B[t_i/x]$ and a substitution θ_i, such that, $H_i = \theta_i(D_i)$. It can also be assumed (using renaming) that the sets of variables occurring in these amplifications are disjoint. It is not difficult to show by induction on the length of a quantifier duplication sequence that for each $B[t_i/x]$ and D_i, there is some renamed copy B_i of B, some amplification D_i' of B_i, and a substitution σ_i, such that, $H_i = \theta_i(\sigma_i(D_i'))$ (σ_i is a substitution that substitutes t_i for renamed occurrences of x). It can also be assumed (using renaming) that the sets of variables occurring in these amplifications are disjoint. Then, note that $D = D_1' \wedge \cdots \wedge D_k'$ is an amplification of A that can be obtained by first applying k quantifier duplications, obtaining $\forall xB \wedge \cdots \wedge \forall xB$ (with k copies of $\forall xB$), and then by amplifying each copy of $\forall xB$ to D_i'. Furthermore, the substitution $\theta = \sigma_1; \theta_1; \cdots; \sigma_k; \theta_k$ is such that $C = \theta(D)$. □

We can now state one of the main theorems of this paper.

THEOREM 5.5. *Given a universal sentence A in nnf, A is unsatisfiable iff some amplification of A has a p-acceptable mating.*

PROOF. First, assume that some amplification D of A has a p-acceptable mating $\mathcal{M}$. Let B be obtained by some sequence of quantifier duplications from A, C the rectified formula equivalent to B, and D the result of deleting quantifiers from C. Let $\forall x_1, \ldots, \forall x_n$ be the quantifiers of C, in the left-to-right order in which they occur in C. It is easy to show that $\models A \equiv B$, $\models B \equiv C$, $\models C \equiv \forall x_1, \ldots, \forall x_n D$, and $\models \forall x_1, \ldots, \forall x_n D \supset \sigma_{\mathcal{M}}(D)$. Since $\mathcal{M}$ is a p-acceptable mating, by Lemma 4.6, it is a refutation mating, and so $\sigma_{\mathcal{M}}(D)$ is unsatisfiable. Hence, A is also unsatisfiable.

Now, suppose that A is unsatisfiable. By the Skolem–Herbrand–Gödel theorem (Theorem 5.2), there is some c-instance C of A which is unsatisfiable. By Lemma 5.4, there is some amplification D of A and a substitution θ such that $C = \theta(D)$. By Lemma 4.6, since $\theta(D)$ is unsatisfiable and D is quantifier free, there is some p-acceptable mating $\mathcal{M}$ for D. □

The following example illustrates Theorem 5.5.

Example 5.6. Let A be the following (equational) sentence:

$$\forall x\, \forall y\, \forall z(*(x, *(y, z)) \doteq *(*(x, y), z))) \tag{1}$$
$$\wedge\ \forall u(*(u, 1) \doteq u) \tag{2}$$
$$\wedge\ \forall v(*(1, v) \doteq v) \tag{3}$$
$$\wedge\ \forall w(*(w, w) \doteq 1) \tag{4}$$
$$\wedge\ \neg(*(a, b) \doteq *(b, a)). \tag{5}$$

The first three equations are the axioms for monoids (a binary operation $*$ which is associative and has an identity element 1), the fourth equation asserts that the square of every element is the identity, and the fifth asserts the negation of the commutativity of $*$ (A is the result of a Skolemization). The unsatisfiability of A asserts that any monoid such that the square of every element is the identity is commutative.

Consider the following amplification D of A and the set MS consisting of one set of literals:

$$\begin{aligned}
D = {} & (*(u_1, 1) \doteq u_1) \\
& \wedge (*(w_1, w_1) \doteq 1) \\
& \wedge (*(x_1, *(y_1, z_1)) \doteq *(*(x_1, y_1), z_1))) \\
& \wedge (*(x_2, *(y_2, z_2)) \doteq *(*(x_2, y_2), z_2))) \\
& \wedge (*(w_2, w_2) \doteq 1) \\
& \wedge (*(1, v_1) \doteq v_1) \\
& \wedge (*(x_3, *(y_3, z_3)) \doteq *(*(x_3, y_3), z_3))) \\
& \wedge (*(x_4, *(y_4, z_4)) \doteq *(*(x_4, y_4), z_4))) \\
& \wedge (*(w_3, w_3) \doteq 1) \\
& \wedge \neg(*(a, b) \doteq *(b, a)).
\end{aligned}$$

$$\begin{aligned}
MS = {} & \{\{(*(u_1, 1) \doteq u_1), \\
& (*(w_1, w_1) \doteq 1), \\
& (*(x_1, *(y_1, z_1)) \doteq *(*(x_1, y_1), z_1))), \\
& (*(x_2, *(y_2, z_2)) \doteq *(*(x_2, y_2), z_2))), \\
& (*(w_2, w_2) \doteq 1), \\
& (*(1, v_1) \doteq v_1), \\
& (*(x_3, *(y_3, z_3)) \doteq *(*(x_3, y_3), z_3))), \\
& (*(x_4, *(y_4, z_4)) \doteq *(*(x_4, y_4), z_4))), \\
& (*(w_3, w_3) \doteq 1), \\
& \neg(*(a, b) \doteq *(b, a))\}\}.
\end{aligned}$$

Let θ be the substitution

$$[a/u_1, (a*b)/w_1, a/x_1, (a*b)/y_1, (a*b)/z_1,$$
$$a/x_2, a/y_2, b/z_2, a/w_2, b/v_1,$$
$$b/x_3, (a*b)/y_3, b/z_3, a/x_4, b/y_4, b/z_4, b/w_3].$$

We claim that $\langle MS, \theta \rangle$ is a mating for D. For simplicity of notation let us adopt infix notation, and denote $*(s,t)$ as $s*t$. Then, we have:

$$\begin{aligned}
a*b &= \{a*1\}*b && \text{by (2)}\\
&= \{\alpha*[(a*b)*(a*b)]\}*b && \text{by (4)}\\
&= \{[a*(a*b)]*(a*b)\}*b && \text{by (1)}\\
&= \{[(a*a)*b]*(a*b)\}*b && \text{by (1)}\\
&= \{[1*b]*(a*b)\}*b && \text{by (4)}\\
&= \{b*(a*b)\}*b && \text{by (3)}\\
&= b*\{(a*b)*b\} && \text{by (1)}\\
&= b*\{a*(b*b)\} && \text{by (1)}\\
&= b*\{a*1\} && \text{by (4)}\\
&= b*a, && \text{by (2)}
\end{aligned}$$

which shows that $\langle MS, \theta \rangle$ is a p-acceptable mating for D (there is a single vertical path in D). Note that eq. (2) instantiated by the substitution $[a/u_1]$ is used twice.

Theorem 5.5 suggests a procedure for showing that a universal sentence in nnf is unsatisfiable: Compute incrementally amplifications of D, and at each stage, test whether such an amplification has a p-acceptable mating. Such a procedure is presented in a later section.

5.3 OUTERMOST QUANTIFIER DUPLICATION. Since the complexity of the search for an acceptable mating grows exponentially as the number of occurrences of literals in the amplification D increases, it is important to keep this number small.

A systematic scheme for duplicating quantifiers that guarantees completeness, is to duplicate *outermost* quantifiers.

Definition 5.7. Given a universal formula A in nnf, a subformula occurrence $\forall xB$ of A is a *maximal quantified subformula* of A iff there is no quantified subformula occurrence $\forall yC$ of A, such that $\forall xB$ is a proper subformula of $\forall yC$.[9] If $\forall xB$ is a maximal quantified subformula occurrence of A, the quantifier $\forall x$ is called an *outermost* quantifier occurrence.

LEMMA 5.8. *Let A be a universal sentence in nnf. Then, A is unsatisfiable iff there is a refutation mating for some amplification D of A, such that, in forming D from A, only outermost quantifier duplications are performed.*

PROOF. It is enough to show that the Skolem–Herbrand–Gödel theorem holds for c-instances obtained as substitution instances of formulas obtained from A by outermost quantifier duplications only. This can be shown in at

[9] For an inductive definition of this concept, see Gallier [17].

least two ways. The first proof is already essentially contained in the proof of Lemma 7.6.2 of Gallier ([17, page 360]). Indeed, this lemma is obtained from Theorem 7.4.1 and Theorem 7.4.2 [17, pages 334 and 337], a Gentzen-like Hauptsatz for a proof system in which quantifier rules apply only to outermost quantifiers (the system $G2^{\mathrm{nnf}}$, [17, page 327]). One simply has to verify that the induction in Lemma 7.6.2 yields the right kind of c-instances, and this is straightforward. The other proof is obtained by observing that the proofs of Lemma 5 and Theorem 2 in Andrews [1, page 208] go through unchanged, as they do not depend on the fact that the language does not contain equality. □

Hence, in searching for a mating, there is no loss of generality in duplicating outermost quantifiers only. However, this is not always the best strategy, and it would be useful to develop heuristics for duplicating quantifiers.

5.4 Comparison With Other Methods. An extension of the method of matings to first-order languages with equality is sketched (without proofs) in Bibel [9, Sect. V.3, pp. 234–242] (under the name *connection method with equality*). Bibel's method uses mated sets similar to ours, except that they are dual to ours, since Bibel's method shows the validity rather than the unsatisfiability of a (Skolemized) sentence. Hence, sets of literals are interpreted as disjunctions. A set of the form

$$\{\neg(s_1 \doteq t_1), \ldots, \neg(s_m \doteq t_m), (s \doteq t)\}$$

is called an *eq-literal*, and a set of the form

$$\{\neg(s_1 \doteq t_1), \ldots, \neg(s_m \doteq t_m), \neg L, L'\}$$

where L and L' are nonequational atoms, is called an *eq-connection*. In our presentation, the use of a many-sorted language with the special sort *bool* allows us to treat a nonequational literal as the special equation $L \doteq \top$, and we only need the first kind of mated set, but this is an inessential detail.

Bibel's method and ours differ significantly in the criterion used for testing the validity (equivalency, unsatisfiability) of a mated set. Bibel defines an eq-literal to be *valid* iff there is some substitution σ such that $\sigma(s_i) = \sigma(t_i)$ for all i, $1 \leq i \leq m$, and $\sigma(s) = \sigma(t)$. An eq-connection is said to be *complementary* iff there is some substitution σ such that $\sigma(s_i) = \sigma(t_i)$ for all i, $1 \leq i \leq m$, and $\sigma(L) = \sigma(L')$.

It should be noted that the notion of a substitution used by Bibel is highly nonstandard. Bibel [9, Sect. III.1.6, page 66] defines a substitution σ as a set of pairs $\{s_1/t_1, \ldots, s_n/t_n\}$, where each t_i is a term to be substituted for s_i, but where s_i itself can be a nonvariable term! Of course, substitutions are applied in a homomorphic fashion, but with this definition, a substitution is *not* necessarily defined on all terms.

To be completely accurate, with Bibel's definition of a substitution, the substitution σ mentioned in the definition of a valid eq-literal is such that it consists of pairs of the form s_i/t_i or t_i/s_i. Then, Theorem V.3.6.C (page 237) states (in our language) that a formula F is valid iff for some amplification D of F, there is a spanning set W of eq-literals and eq-connections and some substitution σ such that, for every eq-literal $w \in W$, $\sigma(w)$ is valid, and for every eq-connection $w \in W$, $\sigma(w)$ is complementary.

This theorem does hold, provided that we allow eq-literals and eq-connections in the set W to contain extra literals arising from instances of the equality axioms. Hence, Bibel's method uses standard unification, but the mated sets may have to include extra literals corresponding to instances of the equality axioms.

In our method, we require that there be some substitution σ such that

$$\{\sigma(s_1 \doteq t_1), \ldots, \sigma(s_m \doteq t_m), \neg\, \sigma(u \doteq v)\}$$

is unsatisfiable, or equivalently, treating the equations in $\sigma(E)$ as ground equations, that $\sigma(u) \overset{*}{\leftrightarrow}_{\sigma(E)} \sigma(v)$ holds. Hence, σ is a special kind of E-unifier (a rigid E-unifier), but there is *no need* to include extra literals corresponding to instances of the equality axioms to our mated sets. The following example should illustrate this point clearly. Consider the eq-literal

$$\left\{\neg\left(f^3a \doteq a\right), \neg\left(f^5a \doteq a\right), \left(fa \doteq a\right)\right\}.$$

It is valid, but yet, there is no substitution in the sense of Bibel demonstrating that it is valid. The only way to show validity is to add additional equality axioms to show that *fa* and *a* are *congruent* modulo the set of equations $\{(f^3a \doteq a), (f^5a \doteq a)\}$. Hence, in Bibel's method, this mated set would have to be expanded before it is shown to be valid. In our method, it would be found valid immediately (actually, its negation would be found unsatisfiable).

Hence, Bibel's method and ours differ in the type of unification and the methods used to check the validity (or unsatisfiability) of mated sets.

In Chapter 4 of his Ph.D. dissertation, Pfenning [38] presents a method for dealing with equality in a system of expansion proofs that involves matings. Pfenning's system applies to higher-order logic, and equality is treated as a defined symbol ($(A \doteq B)$ is an abbreviation for $\forall Q(Q(A) \supset Q(B))$, where Q is a predicate variable). As pointed out by Pfenning, it is theoretically possible to derive the mated sets arising in our method from the mated sets used in his method via the translation mentioned above. In some sense, our way of checking mated sets is an optimization of Pfenning's method restricted to the first-order case. However, it does not seem possible to obtain the completeness of our method in this fashion. Our method is also different in a more radical sense, which is that Pfenning's method uses higher-order unification, whereas we use a special form of E-unification that is decidable. This suggests that there may be a form of rigid higher-order unification, but we have not explored this possibility.

6. *Complete Sets of Rigid E-Unifiers*

We have already noted in Remark (2) after Definition 4.5 that $\mathcal{M} = \langle MS, \sigma \rangle$ is an equational mating iff σ is a rigid $\vec{E}$-unifier of S, where $\vec{E} = (E_{\mathscr{S}})_{\mathscr{S} \in MS}$ and $S = \{\langle s, t \rangle \mid \mathscr{S} \in MS\}$, the family of sets of equations and the set of pairs associated with the mated sets $\mathscr{S} = \{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m), \neg(s \doteq t)\} \in MS$. It is obviously crucial to show that there is an algorithm for testing whether a family of mated sets forms a mating. From the above observation, this is equivalent to deciding whether a pair $\langle \vec{E}, S \rangle$ is an equational premating. In the following sections, it will be shown that this problem is NP-complete. Actually, this result is an easy extension of a simpler problem, and we now focus on this problem.

Problem. Given a finite set $E = \{u_1 \doteq v_1, \ldots, u_n \doteq v_n\}$ of equations and a pair $\langle u, v \rangle$ of terms, is there a substitution θ such that, treating $\theta(E)$ as a set of ground equations, $\theta(u) \doteq_{\theta(E)} \theta(v)$, that is, $\theta(u)$ and $\theta(v)$ are congruent modulo $\theta(E)$ (by congruence closure)?

The substitution θ is called a *rigid E-unifier of u and v.*

It is interesting to observe, as pointed out by Jean Yves Girard, that the notion of rigid E-unification arises by *bounding* the resources, in this case, the number of available instances of equations from E. To be precise, only a single instance of each equation in E can be used, and in fact, these instances $\theta(u_1 \doteq v_1), \ldots, \theta(u_n \doteq v_n)$ must arise from the *same* substitution θ. Also, once these instances have been created, the remaining variables (if any) are considered rigid, that is, treated as constants, so that it is not possible to instantiate these instances. This is in the spirit of linear logic [23]. The special case of rigid E-unification where E is a set of ground equations has been investigated by Kozen who has shown that this problem is NP-complete (Kozen [30, 31]). Thus, rigid E-unification is NP-hard. We show that it is also in NP, and hence it is NP-complete.

Suppose we want to find a rigid E-unifier θ of u and v. Roughly, the idea is to use a form of unfailing completion procedure (Knuth and Bendix [29], Huet [27], Bachmair [4], Bachmair et al. [5, 6]). In order to clarify the differences between our method and unfailing completion, especially for readers unfamiliar with this method, we briefly describe the use of unfailing completion as a refutation procedure. For more details, the reader is referred to Bachmair [4].

Let E be a set of equations, and $\succ$ a reduction ordering total on ground terms. The central concept is that of E being *ground Church–Rosser with respect to* $\succ$. The crucial observation is that every ground instance $\sigma(l) \doteq \sigma(r)$ of an equation $l \doteq r \in E$ is orientable with respect to $\succ$, since $\succ$ is total on ground terms. Let $E^{\succ}$ be the set of all instances $\sigma(l) \doteq \sigma(r)$ of equations $l \doteq r \in E \cup E^{-1}$ with $\sigma(l) \succ \sigma(r)$ (the set of *orientable instances*). We say that E is *ground Church–Rosser with respect to* $\succ$ iff for every two ground terms u, v, if $u \overset{*}{\leftrightarrow}_E v$, then there is some ground term w such that $u \overset{*}{\rightarrow}_{E^{\succ}} w$ and $w \overset{*}{\leftarrow}_{E^{\succ}} v$. Such a proof is called a *rewrite proof.*

An unfailing completion procedure attempts to produce a set E^{∞} equivalent to E and such that E^{∞} is ground Church–Rosser with respect to $\succ$. In other words, every ground equation provable from E has a rewrite proof in E^{∞}. The main mechanism involved is the computation of critical pairs. Given two equations $l_1 \doteq r_1$ and $l_2 \doteq r_2$ where l_2 is unifiable with a subterm l_1/β of l_1 which is not a variable, the pair $\langle \sigma(l_1[\beta \leftarrow r_2]), \sigma(r_1) \rangle$ where σ is a mgu of l_1/β and l_2 is a *critical pair.*

If we wish to use an unfailing completion procedure as a refutation procedure, we add two new constants T and F and a new binary function symbol eq to our language. In order to prove that $E \vdash u \doteq v$ for a ground equation $u \doteq v$, we apply the unfailing completion procedure to the set $E \cup \{eq(u, v) \doteq F, eq(z, z) \doteq T\}$, where z is a new variable. It can be shown that $E \vdash u \doteq v$ iff the unfailing completion procedure generates the equation $F \doteq T$. Basically, given any proof of $F \doteq T$, the unfailing completion procedure extends E until a rewrite proof is obtained. It can be shown that unfailing completion is a complete refutation procedure, but of course, it is not a decision procedure. It should also be noted that when unfailing completion is used as a refutation

procedure, E^{∞} is actually never generated. It is generated "by need," until $F \doteq T$ turns up.

We now come back to our situation. Without loss of generality, it can be assumed that we have a rigid E-unifier θ of T and F such that $\theta(E)$ is ground. In this case, equations in $\theta(E)$ are orientable instances. The crucial new idea is that in trying to obtain a rewrite proof of $F \doteq T$, we still compute critical pairs, but we *never rename variables*. If l_2 is equal to l_1/β, then we get a critical pair essentially by simplification. Otherwise, some variable in l_1 or in l_2 gets bound to a term *not* containing this variable. Thus, the total number of variables in E keeps decreasing. Therefore, after a polynomial number of steps (in fact, the number of variables in E) we must stop or fail. So we get membership in NP. Oversimplifying a bit, we can say that our method is a form of lazy unfailing completion with no renaming of variables.

However, there are some significant departures from traditional Knuth–Bendix completion procedures, and this is for two reasons. The first reason is that we must ensure termination of the method. The second is that we want to show that the problem is in NP, and this forces us to be much more concerned about efficiency.

The proof that rigid E-unification is in NP requires quite a bit of machinery, and since this paper is already long, we focus on the algorithmic aspect of the result, leaving out most proofs. Full details can be found in Gallier et al. [22].

In order to show that our decision procedure is in NP, we need the fact that if two terms u and v are unifiable, a most general unifier (mgu) of u and v can be represented concisely in triangular form (the size of this system is linear in the number of symbols in u and v). This result can be obtained from the fast method using multiequations of Martelli and Montanari [33] or the fast method using the graph unification closure of Paterson and Wegman [37].

Definition 6.1. A *term pair* (or *pair*) is just a pair of two terms, denoted by $\langle s, t \rangle$, and a substitution θ is called a *unifier* of a pair $\langle s, t \rangle$ if $\theta(s) = \theta(t)$. A *term system* (or *system*) is a set of such pairs, and a substitution θ is a unifier of a system if it unifies each pair. A substitution σ is an (*idempotent*) *most general unifier*, or *mgu*, of a system S iff (i) $D(\sigma) \subseteq Var(S)$ and $D(\sigma) \cap I(\sigma) = \emptyset$ (σ is idempotent); (ii) σ is a unifier of S; (iii) For every unifier θ of S, $\sigma \leq \theta$ (where $\sigma \leq \theta$ iff $\theta = \sigma; \eta$ for some η).

Definition 6.2. Given an idempotent substitution σ (i.e., $D(\sigma) \cap I(\sigma) = \emptyset$) with domain $D(\sigma) = \{x_1, \ldots, x_k\}$, a *triangular form* for σ is a finite set T of pairs $\langle x, t \rangle$ where $x \in D(\sigma)$ and t is a term, such that this set T can be sorted (possibly in more than one way) into a sequence $\langle \langle x_1, t_1 \rangle, \ldots, \langle x_k, t_k \rangle \rangle$ satisfying the following properties: for every i, $1 \leq i \leq k$,

(1) $\{x_1, \ldots, x_i\} \cap Var(t_i) = \emptyset$, and
(2) $\sigma = [t_1/x_1]; \cdots; [t_k/x_k]$.

The set of variables $\{x_1, \ldots, x_k\}$ is called the *domain* of T. Note that in particular $x_i \notin Var(t_i)$ for every i, $1 \leq i \leq k$, but variables in the set $\{x_{i+1}, \ldots, x_k\}$ may occur in $t_1, \ldots, t_i$. It is easily seen that σ is an (idempotent) mgu of the term system T.

Example 6.3. Consider the substitution $\sigma = [f(f(x_3, x_3), f(x_3, x_3))/x_1, f(x_3, x_3)/x_2]$. The system $T = \{\langle x_1, f(x_2, x_2) \rangle, \langle x_2, f(x_3, x_3) \rangle\}$ is a triangu-

lar form of σ since it can be ordered as $\langle\langle x_1, f(x_2, x_2)\rangle, \langle x_2, f(x_3, x_3)\rangle\rangle$ and $\sigma = \{f(x_2, x_2)/x_1\}; [f(x_3, x_3)/x_2]$.

The triangular form $T = \{\langle x_1, t_1\rangle, \ldots, \langle x_k, t_k\rangle\}$ of a substitution σ also defines a substitution; namely, $\sigma_T = [t_1/x_1, \ldots, t_k/x_k]$. This substitution is usually different from σ and not idempotent as can be seen from Example 6.3. However, this substitution plays a crucial role in our decision procedure because of the following property.

LEMMA 6.4. *Given a triangular form* $T = \{\langle x_1, t_1\rangle, \ldots, \langle x_k, t_k\rangle\}$ *for a substitution* σ *and the associated substitution* $\sigma_T = [t_1/x_1, \ldots, t_k/x_k]$, *for every unifier* θ *of* T, $\theta = \sigma_T; \theta$.

Another important observation about σ_T is that even though it is usually not idempotent, at least one variable in $\{x_1, \ldots, x_k\}$ does not belong to $I(\sigma_T)$ (otherwise, condition (1) of the triangular form fails). We assume that a procedure TU is available, which, given any unifiable term system S, returns a triangular form for an idempotent mgu of S, denoted by $TU(S)$. When S consists of a single pair $\langle u, v\rangle$, $TU(S)$ is also denoted by $TU(u, v)$.

Recall that we write $u \doteq_E v$ to express that $u \leftrightarrow_E v$, treating the equations in E as *ground equations*.

Definition 6.5. Let $E = \{(s_1 \doteq t_1), \ldots, (s_m \doteq t_m)\}$ be a finite set of equations, and let $Var(E) = \bigcup_{(s=t)\in E} Var(s \doteq t)$ denote the set of variables occurring in E.[10] Given a substitution θ, we let $\theta(E) = \{\theta(s_i \doteq t_i) \mid s_i \doteq t_i \in E, \theta(s_i) \neq \theta(t_i)\}$. Given any two terms u and v,[11] a substitution θ is a *rigid unifier of* u *and* v *modulo* E (for short, a *rigid* E*-unifier of* u *and* v) iff

$\theta(u) \doteq_{\theta(E)} \theta(v)$, that is, $\theta(u)$ and $\theta(v)$ are congruent modulo the set $\theta(E)$ considered as a set of *ground equations*.

Definition 6.6. Let E be a (finite) set of equations, and W a (finite) set of variables. For any two substitutions σ and θ, $\sigma =_E \theta[W]$ iff $\sigma(x) \doteq_E \theta(x)$ for every $x \in W$. The relation $\sqsubseteq_E$ is defined as follows. For any two substitutions σ and θ, $\sigma \sqsubseteq_E \theta[W]$ iff $\sigma =_{\theta(E)} \theta[W]$. The set W is omitted when $W = X$ (where X is the set of variables), and similarly E is omitted when $E = \emptyset$.

Intuitively speaking, $\sigma \sqsubseteq_E \theta$ iff σ can be generated from θ using the equations in $\theta(E)$. Clearly, $\sqsubseteq_E$ is reflexive. However, it is not symmetric as shown by the following example.

Example 6.7. Let $E = \{fx \doteq x\}$, $\sigma = [fa/x]$ and $\theta = [a/x]$. Then, $\theta(E) = \{fa \doteq a\}$ and $\sigma(x) = fa \doteq_{\theta(E)} a = \theta(x)$, and so $\sigma \sqsubseteq_E \theta$. On the other hand, $\sigma(E) = \{ffa \doteq fa\}$, but a and fa are not congruent from $\{ffa \doteq fa\}$. Thus, $\theta \sqsubseteq_E \sigma$ *does not* hold.

It is not difficult to show that $\sqsubseteq_E$ is also transitive. We also need an extension of $\sqsubseteq_E$ defined as follows.

Definition 6.8. Let E be a (finite) set of equations, and W a (finite) set of variables. The relation $\leq_E$ is defined as follows: for any two substitutions σ and θ, $\sigma \leq_E \theta[W]$ iff $\sigma;\eta \sqsubseteq_E \theta[W]$ for some substitution η (that is, $\sigma;\eta =_{\theta(E)} \theta$ for some η).

[10] It is possible that equations have variables in common.

[11] It is possible that u and v have variables in common with the equations in E.

Intuitively speaking, $\sigma \leq_E \theta$ iff σ is more general than some substitution that can be generated from θ using $\theta(E)$. Clearly, $\leq_E$ is reflexive. The transitivity of $\leq_E$ is also shown easily. When $\sigma \leq_E \theta[W]$, we say that σ *is (rigid) more general than* θ *over* W. It can be shown that if σ is a rigid E-unifier of u and v and $\sigma \leq_E \theta$, then θ is a rigid E-unifier of u and v. The converse is false.

Definition 6.9. Given a (finite) set E of equations, for any two terms u and v, letting $V = Var(u) \cup Var(v) \cup Var(E)$, a set U of substitutions is a *complete set of rigid E-unifiers for u and v* iff: For every $\sigma \in U$,

(i) $D(\sigma) \subseteq V$ and $D(\sigma) \cap I(\sigma) = \varnothing$ (idempotence),
(ii) σ is a rigid E-unifier of u and v,
(iii) For every rigid E-unifier θ of u and v, there is some $\sigma \in U$, such that, $\sigma \leq_E \theta[V]$.

It is very useful to observe that if a procedure P for finding sets of rigid E-unifiers satisfies the property stated in Definition 6.10 given next, then in order to show that this procedure yields complete sets, there is no loss of generality in showing completeness with respect to *ground* rigid E-unifiers whose domains contain V (i.e., in clause (iii) of Definition 6.9, $\theta(x)$ is a ground term for every $x \in D(\theta)$, and $V \subseteq D(\theta)$).

Definition 6.10. A procedure P for finding sets of rigid E-unifiers is *pure* iff the following condition holds: For every ranked alphabet Σ, every finite set E of equations over $T_\Sigma(X)$ and every $u, v \in T_\Sigma(X)$, if $U = P(E, u, v)$ is the set of rigid E-unifiers for u and v given by procedure P, then for every $\sigma \in U$, for every $x \in D(\sigma)$, every constant or function symbol occurring in $\sigma(x)$ occurs either in some equation in E or in u or in v.

In other words, $P(E, u, v)$ does not contain constant or function symbols that do not already occur in the input (E, u, v). To prove what we claimed, we proceed as follows. We add countably infinitely many new (distinct) constants c_x to Σ, each constant c_x being associated with the variable x. The resulting alphabet is denoted by Σ_{SK}. If θ is not ground, we create the Skolemized version of θ, that is, the substitution $\hat{\theta}$ obtained by replacing the variables in the terms $\theta(x)$ by new (distinct) constants.[12]

LEMMA 6.11. Given a rigid E-unification procedure P satisfying the property of Definition 6.10, assume that for every ranked alphabet Σ, every finite set E of equations over $T_\Sigma(X)$ and every $u, v \in T_\Sigma(X)$, the set $U = P(E, u, v)$ of rigid E-unifiers of u and v given by P satisfies conditions (i) and (ii) of Definition 6.9, and the new condition (iii′): for every rigid E-unifier θ of u and v such that $V \subseteq D(\theta)$ and $\theta(x) \in T_\Sigma$ for every $x \in D(\theta)$, there is some $\sigma \in U$ such that $\sigma \leq_E \theta[V]$ (where $V = Var(E) \cup Var(u, v)$). Then every set $U = P(E, u, v)$ is a complete set of rigid E-unifiers for u and v.

7. *Minimal Rigid E-Unifiers*

One of the reasons for the decidability of rigid E-unification is that if a pair $\langle u, v \rangle$ has some rigid E-unifier, then it has a rigid E-unifier that is minimal in a sense made precise in the sequel. Given a finite or countably infinite ranked

[12] That is, $\hat{\theta}$ is obtained from θ by replacing every variable y in each term $\theta(x)$ by the corresponding Skolem constant c_y, for each $x \in D(\theta)$.

alphabet Σ, it is always possible to define a total simplification ordering $\preceq$ on T_Σ (the set of all ground terms) [11]. We use the total simplification ordering $\prec$ on T_Σ to define a well-founded partial order $\ll$ on ground substitutions. For this, it is assumed that the set of variables X is totally ordered as $X = \langle x_1, x_2, \ldots, x_n, \ldots \rangle$.

Definition 7.1. The partial order $\ll$ is defined on ground substitutions as follows. Given any two ground substitutions σ and θ such that $D(\sigma) = D(\theta)$, letting $\langle y_1, \ldots, y_n \rangle$ be the sequence obtained by ordering the variables in $D(\sigma)$ according to their order in X, then $\sigma \ll \theta$ iff

$$\langle \sigma(y_1), \ldots, \sigma(y_n) \rangle \preceq_{lex} \langle \theta(y_1), \ldots, \theta(y_n) \rangle,$$

where $\preceq_{lex}$ is the lexicographic ordering on tuples induced by $\preceq$.

Since $\preceq$ is well-founded and $\ll$ is induced by the lexicographic ordering $\preceq_{lex}$ which is well-founded, $\ll$ is also well-founded. In fact, given any finite set V of variables, note that $\ll$ is a total well-founded ordering for the set of ground substitutions with domain V.

Given a set E of equations and a total simplification ordering $\preceq$ on ground terms, for any ground substitution θ, we let $\theta(E)$ denote the set $\{\theta(l) \doteq \theta(r) \mid \theta(l) \succ \theta(r),\ l \doteq r \in E \cup E^{-1}\}$ of oriented instances of E. Thus, we can also view $\theta(E)$ as a set of rewrite rules.

The reason for considering the well-founded ordering $\ll$ on ground substitutions is that minimal rigid E-unifiers exist. This is one of the reasons for the decidability of rigid E-unification. The example below gives some motivation for the next definition and lemma.

Example 7.2. Let $E = \{fa \doteq a, x \doteq fa\}$, and $\langle u, v \rangle = \langle gx, x \rangle$. It is obvious that there is a simplification ordering total on ground terms such that $a \prec f \prec g$ (for instance, a *recursive path ordering*, see Dershowitz [11]). The main point of this example is the fact that some rigid E-unifiers of gx and x are redundant, in the sense that they are subsumed by rigid E-unifiers that are smaller with respect to $\leq_E$. For instance, $\theta = [gf^{10}a/x]$ is a rigid E-unifier of gx and x, but so is $\sigma = [ga/x]$, and $\sigma \sqsubseteq_E \theta$.

An illustration of the redundancy of θ is the fact that $\theta(x) = gf^{10}a$ is reducible by the rule $fa \to a$. The fact that some term $\theta(x)$ may be reducible by some oriented instance $\theta(l) \to \theta(r)$ of an equation $l \doteq r \in E \cup E^{-1}$ turns out to be a problem for the completeness of the method. In order to avoid such redundancies, for every rigid E-unifier θ of u and v, we consider the set $S_{E,u,v,\theta}$ of all ground rigid E-unifiers ρ of u and v such that $\rho \sqsubseteq_E \theta$. The crucial fact is that the set $S_{E,u,v,\theta}$ has a smallest element σ under the ordering $\ll$, and that this least substitution is nicely reduced with respect to $\sigma(E)$. Intuitively speaking, we find the least ground rigid E-unifier σ of u and v constructible from θ and $\theta(E)$ (least with respect to $\ll$). Referring back to $\theta = [gf^{10}a/x]$, the substitution $\sigma = [ga/x]$ is the smallest element of $S_{E,u,v,\theta}$. It is not sufficient to simply consider all ground substitutions ρ such that $\rho \sqsubseteq_E \theta$, because some of them may not be rigid E-unifiers of u and v. For instance, we have $\rho \sqsubseteq_E \theta$ for $\rho = [a/x]$, but ρ is not a rigid E-unifier of ga and a since $\rho(E) = \{fa \doteq a\}$. Thus, we have to consider rigid E-unifiers of u and v such that $\rho \sqsubseteq_E \theta$.

The least element σ of the set $S_{E,u,v,\theta}$ enjoys some nice reduction properties with respect to $\sigma(E)$. These properties stated in the forthcoming lemma will be used in the proof that the method is complete.

Definition 7.3. Let E be a set of equations (over $T_\Sigma(X)$) and $u, v \in T_\Sigma(X)$ any two terms. For any ground rigid E-unifier θ of u and v, let

$$S_{E,u,v,\theta} = \left\{\rho \mid D(\rho) = D(\theta), \rho(u) \doteq_{\rho(E)} \rho(v), \rho \sqsubseteq_E \theta, \text{and } \rho \text{ ground}\right\}.$$

Obviously, $\theta \in S_{E,u,v,\theta}$, so $S_{E,u,v,\theta}$ is not empty. Since $\lessdot\!\lessdot$ is total and well-founded on ground substitutions with domain $D(\theta)$, the set $S_{E,u,v,\theta}$ contains some least element σ (with respect to $\lessdot\!\lessdot$).

We shall now prove the following crucial result: For this, recall that a *degenerate equation* is of the form $x \doteq t$, where x is a variable and $x \notin Var(t)$, and that a *nondegenerate equation* is an equation that is not degenerate.

LEMMA 7.4. *Let E be a set of equations (over $T_\Sigma(X)$) and $u, v \in T_\Sigma(X)$ any two terms. For any ground rigid E-unifier θ of u and v, if σ is the least element of the set $S_{E,u,v,\theta}$ of Definition* 7.3, *then the following properties hold*:

(1) *Every term of the form $\sigma(x)$ is irreducible by every oriented instance $\sigma(l) \to \sigma(r)$ of a nondegenerate equation $l \doteq r \in E \cup E^{-1}$, and*
(2) *Every proper subterm of a term of the form $\sigma(x)$ is irreducible by every oriented instance $\sigma(l) \to \sigma(r)$ of a degenerate equation $l \doteq r \in E \cup E^{-1}$.*

8. *The Reduction Procedure*

One of the major components of the decision procedure for rigid E-unification is a procedure for creating a reduced set of rewrite rules equivalent to a given (finite) set of ground equations. This procedure first presented in Gallier et al. [19] runs in polynomial time. However, due to the possibility that variables may occur in the equations, we have to make some changes to this procedure. Roughly speaking, given a "guess" $\mathcal{O}$ (which we call an *order assignment*) of the ordering among all subterms of the terms in a set of equations E, we can run the reduction procedure R on E and $\mathcal{O}$ to produce a reduced rewrite system $R(E, \mathcal{O})$ equivalent to E, and whose orientation is dictated by the ordering $\mathcal{O}$.

Definition 8.1. Given a set R of rewrite rules, we say that R *is rigid reduced* iff

(1) No lefthand side of any rewrite rule $l \to r \in R$ is reducible by any rewrite rule in $R - \{l \to r\}$ treated as a ground rule;
(2) No righthand side of any rewrite rule $l \to r \in R$ is reducible by any rewrite rule in R treated as a ground rule.

Definition 8.2. Given two sets E and E' of equations, we say that E and E' are *rigid equivalent* iff for every two terms u and v, $u \doteq_E v$ iff $u \doteq_{E'} v$ (treating E and E' as sets of ground equations).

For technical reasons, it will be convenient to view the problem of rigid E-unification as the problem of deciding whether two fixed constants are rigid E-unifiable. This can be achieved as follows (the idea is borrowed from Dershowitz). Let eq be a new binary function symbol not occurring in Σ, and T

and F two new constants not occurring in Σ. The following simple but useful lemma holds.

LEMMA 8.3. *Given a set E of equations and any two terms u and v, a substitution θ over $T_\Sigma(X)$ is a rigid E-unifier of u and v iff there is some substitution θ' over $T_\Sigma(X)$ such that $\theta = \theta'|_{D(\theta')-\{z\}}$ and $T \overset{\cdot}{=}_{\theta'(E_{u,v})} F$, where $E_{u,v} = E \cup \{eq(u,v) \doteq F, eq(z,z) \doteq T\}$, and z is a new variable not in $Var(E) \cup Var(u,v)$.*

The total simplification ordering $\preceq$ can be extended to the set

$$T_\Sigma \cup \{T, F\} \cup \{eq(u,v) \mid u, v \in T_\Sigma\}.$$

For details, see [22]. We need to show that in searching for rigid E-unifiers, it is always possible to deal with sets of equations that are rigid reduced. The proof of this fact uses the result shown elsewhere that every finite set E of ground equations is equivalent to a reduced set $R(E)$ of rewrite rules. We now review the procedure first presented in Gallier et al. [19], which, given a total simplification ordering $\prec$ on ground terms and a finite set E of ground equations returns a reduced rewrite system $R(E)$ equivalent to E.

Definition 8.4 (*Basic reduction procedure*). Let E be a finite set of ground equations, and $\prec$ a simplification ordering total on ground terms. The basic reduction procedure generates a finite sequence of triples $\langle \mathscr{E}_i, \Pi_i, \mathscr{R}_i \rangle$ where $\mathscr{E}_i$ is a finite set of ground equations, Π_i is a partition (associated with $\mathscr{E}_i$), and $\mathscr{R}_i$ is a set of ground rewrite rules. Given a triple $\langle \mathscr{E}_i, \Pi_i, \mathscr{R}_i \rangle$, we let $\mathscr{T}_i$ be the set of all subterms of terms occurring in equations in $\mathscr{E}_i$ or in rewrite rules in $\mathscr{R}_i$. The procedure makes use of the *congruence closure* of a finite set of ground equations (Kozen [30, 31], Nelson and Oppen [36], Downey et al. [13]). Congruence closures are represented by their associated partition Π. Given an equivalence relation represented by its partition Π, the equivalence class of t is denoted by $[t]_\Pi$, or $[t]$. Recall that s, t are in the same equivalence class of Π iff s and t are subterms of the terms occurring in E and $s \overset{*}{\leftrightarrow}_E t$ (for details, see Gallier [17]). The congruence closure algorithm will only be run once on E to obtain Π_0, but the partition Π_i may change due to further steps (simplification steps).

begin algorithm

Initially, we set $\mathscr{E}_0 = E, \mathscr{R}_0 = \emptyset$, and run a congruence closure algorithm on the ground set E to obtain Π_0. $i := 0$;

while Π_i has some nontrivial equivalence class[13] **do** {Simplification steps}

Let ρ_{i+1} be the smallest element[14] of the set

$$\bigcup_{C \in \Pi_i, |C| \geq 2} C$$

of terms belonging to nontrivial classes in Π_i.[15] Let C_{i+1} be the nontrivial class that contains ρ_{i+1}, and write $C_{i+1} = \{\rho_{i+1}, \lambda_{i+1}^1, \ldots, \lambda_{i+1}^{k_{i+1}}\}$, where $k_{i+1} \geq 1$, since C_{i+1} is nontrivial. Let $\mathscr{S}_{i+1} = \{\lambda_{i+1}^1 \to \rho_{i+1}, \ldots, \lambda_{i+1}^{k_{i+1}} \to \rho_{i+1}\}$.

{Next, we use the rewrite rules in $\mathscr{S}_{i+1}$ to simplify the rewrite rules in $\mathscr{R}_i \cup \mathscr{S}_{i+1}$, the partition Π_i, and the equations in $\mathscr{E}_i$.}

[13] That is, a class containing at least two elements, in which case $\mathscr{E}_i$ has at least one nontrivial equation.

[14] In the ordering $\prec$.

[15] Where $|C|$ denotes the cardinality of the set C.

To get $\mathscr{R}_{i+1}$, first, we get a canonical system equivalent to $\mathscr{S}_{i+1}$. For this, for every lefthand side λ of a rule in $\mathscr{S}_{i+1}$, replace every maximal redex of λ of the form λ' by ρ, where $\lambda' \to \rho \in \mathscr{S}_{i+1} - \{\lambda \to \rho\}$.[16] Let $\mathscr{S}'_{i+1}$ be the set of simplified rules. Also, let $\mathscr{R}'_{i+1}$ be the set obtained by simplifying the lefthand sides of rules in $\mathscr{R}_i$ using $\mathscr{S}_{i+1}$ (reducing maximal redexes only), and let

$$\mathscr{R}_{i+1} = \mathscr{R}'_{i+1} \cup \mathscr{S}'_{i+1}.$$

Finally, use $\mathscr{S}_{i+1}$ to simplify all terms in Π_i and $\mathscr{E}_i$, using the simplification process described earlier, to obtain Π_{i+1} and $\mathscr{E}_{i+1}$.

$$i := i + 1$$

endwhile

{All classes of Π_i are trivial, and the set $\mathscr{R}_i$ is a canonical system equivalent to E.}

end algorithm

It is shown in [19] that the above procedure always terminates with a system $\mathscr{R}_m$ equivalent to E that is reduced (and hence, canonical).

However, in order to show later that our decision method is in NP, it turns out that we need a sharpening of the above result. We need to show that given a set E of ground equations, the term DAG associated with any equivalent reduced system R is of size no greater than the size of the term DAG associated with E itself, and that the number of rules in R is no greater than the number of equations in E. This is not at all obvious for our algorithm, but fortunately true. To be more specific, the *term DAG* associated with a finite set $\mathscr{S}$ of terms is the labeled directed graph whose set of nodes is the set of all subterms occurring in terms in $\mathscr{S}$, where every constant symbol c or variable x is a terminal node labeled with c or x, and where every node $f(t_1, \ldots, t_k)$ is labeled with f and has exactly the k nodes $t_1, \ldots, t_k$ as immediate successors. In the case of a set of equations (or rewrite rules), the set of terms under consideration is the set of subterms occurring in lefthand or righthand sides of equations (or rules). If a term DAG has m edges and n nodes, we define its size as (m, n).

The quickest way to prove this sharper result is to appeal to two facts:

The first one is due to Metivier [34] and Dershowitz et al. [12] (in fact, a direct proof is quite short).

LEMMA 8.5. *If R and R' are two equivalent reduced rewriting systems contained in some reduction ordering $\succ$, then $R = R'$.*

The second fact is that given a set E of p ground equations with term DAG of size (m, n), a reduced equivalent system R of p' rules with term DAG of size (m', n') such that $m' \leq m$, $n' \leq n$, and $p' \leq p$, is produced by a reduction process that is essentially just a Knuth–Bendix procedure restricted to ground terms.

Definition 8.6. Let $\succ$ be a reduction ordering total on ground terms. Let R be a multiset of oriented pairs (s, t) that we may denote by $s \to t$ of $s \succ t$ and $s \leftarrow t$ if $s \prec t$. Finally, let $\to_R$ denote the rewriting relation induced by the

[16] By a maximal redex of λ, we mean a redex of λ that is not a proper subterm of any other redex of λ. The simplified term is irreducible with respect to $\mathscr{S}_{i+1}$, so these replacements are only done once, and they can be done in parallel because they apply to independent subterms of λ.

nontrivial pairs. The first transformation simply removes trivial pairs from R:

$$\{(u,u)\} \cup R \Rightarrow R. \tag{6}$$

The second orients rules:

$$\{s \to t\} \cup R \Rightarrow \{t \to s\} \cup R. \tag{7}$$

Next, if $r \to_R r'$, then

$$\{l \to r\} \cup R \Rightarrow \{l \to r'\} \cup R, \tag{8}$$

and finally, if $l \to_R l'$, then

$$\{l \to r\} \cup R \Rightarrow \{(l',r)\} \cup R. \tag{9}$$

It should be noted that $\cup$ denotes multiset union, which implies that when a transformation is applied, the occurrence of the rule to which it is applied on the lefthand side (for instance, $s \leftarrow t$ in (7)) no longer exists on the righthand side.

We now show that our reduction method always produces reduced systems whose associated term DAG is no greater than the term DAG associated with the input.

THEOREM 8.7. *Let $\succ$ be a simplification ordering total on ground terms. If E is a set of p ground equations, R an equivalent reduced set of p' ground rewrite rules contained in $\succ$, and (m,n) and (m',n') are the sizes of the term DAGs associated with E and R respectively, then $m' \leq m$, $n' \leq n$, and $p' \leq p$.*

PROOF. We prove this by showing that every sequence of transformations issuing from E must eventually terminate with the set R, and that the size inequality stated above holds. Let

$$E = R_0 \Rightarrow R_1 \Rightarrow R_2 \Rightarrow \cdots$$

be any sequence of transformations starting with E and using the given ordering $\succ$. It is tedious but not hard to show that the transformations produce equivalent sets of rules, and we leave this to the reader. Similarly, it is not hard to show that any set that cannot be transformed must be a reduced set of rules contained in $\succ$, since otherwise some transformation would apply. Now, by Lemma 8.5, if such a terminal set exists, it must be unique, and so it will be identical with R. Thus, we next show that the relation $\Rightarrow$ is noetherian.

For any R, let $\mu(R) = \langle M, k \rangle$, where M is the multiset of all terms occurring in pairs in R and k is the number of pairs of the form $s \leftarrow t$. Let the ordering associated with this measure use the multiset extension of $\succ$ for the first component and the standard ordering on the natural numbers for the second. Clearly this ordering is well-founded, since $\succ$ is. But then, each transformation reduces the measure of the set of pairs, since (6), (8), and (9) reduce M, and (7) reduces k without changing M. Thus, any sequence of transformations must eventually terminate in the set R.

Finally, for any transformation $R_i \Rightarrow R_{i+1}$, note that the size of the current term DAG cannot increase, since (6) deletes nodes and possibly edges, (7) does not change the size, and (8) and (9) possibly decrease the number of nodes and preserve the number of edges. As a matter of fact, these transformations can

be implemented by moving pointers. It is also obvious that each transformation either preserves or decreases the total number of rules. Thus, the claim follows by induction on the length of the transformation sequence. □

Another useful fact needed later is that the time complexity of the reduction procedure is in fact bounded by $O((m + n + p)^3)$, where (m, n) is the size of the term DAG associated with the input E, and p is the number of equations in E.

Unfortunately, given a *nonground* set E of equations, the reduction procedure just presented may not be applicable since some of the equivalence classes may contain terms involving variables and the ordering $\prec$ may no longer be total on such a partition. We need to guess how terms containing variables compare to other terms in the partition in order to reduce the equations. However, it is useful to observe that the reduction algorithm applies, as long as at every stage of the algorithm, it is possible to determine the least element of each nontrivial equivalence class and to sort these least elements. This observation shows that in extending a simplification ordering $\prec$ total on ground terms to terms containing variables, it is sufficient to require this extension to have a least element in each nontrivial equivalence class and to be total on the set of least elements of these classes. Definition 8.10 will make use of this fact.

The key to extending ground orderings is that if some ground rigid E-unifier θ exists, since the ordering $\prec$ is total on ground terms, θ induces a preorder on the terms occurring in the congruence closure Π of E. For example, if $E = \{fa \doteq a, fa \doteq x\}$, $u = gx$, $v = x$, and $\theta = [ga/x]$, then Π has a single nontrivial class $\{fa, a, x\}$, and considering the recursive path ordering such that $a \prec f \prec g$ (see Dershowitz [11]), we have $a \prec fa \prec ga = \theta(x)$. Hence, we can extend $\prec$ so that $fa \prec x$. This way, the equations can be oriented as $fa \to a$, $x \to fa$.

We shall define the concept of an order assignment in order to formalize the above intuition. First, we define some relations induced by a ground substitution on a finite set of terms.

Definition 8.8. Given a finite set S of terms, let $ST(S)$ be the set of all subterms of terms in S (including the terms in S). Let $\preceq$ be a total simplification ordering on ground terms, and θ a ground substitution such that $Var(S) \subseteq D(\theta)$. The relations $\equiv_{\theta,S}$ and $\preceq_{\theta,S}$ on $ST(S)$ are defined as follows: For every $u, v \in ST(S)$,

$$u \preceq_{\theta,S} v \quad \text{iff} \quad \theta(u) \preceq \theta(v),$$

and

$$u \equiv_{\theta,S} v \quad \text{iff} \quad \theta(u) = \theta(v).$$

When we have a partition Π induced by the congruence closure of a finite set E of equations treated as ground, S consists of the lefthand sides and righthand sides of equations in E, and we denote $\preceq_{\theta,S}$ as $\preceq_{\theta,\Pi}$ and $\equiv_{\theta,S}$ as $\equiv_{\theta,\Pi}$. As the next example shows, the equivalence relation $\equiv_{\theta,\Pi}$ may be nontrivial.

Example 8.9. Let $E = \{fx \doteq fgy, fgy \doteq gy, hgz \doteq gz\}$, $u = k(fx, gb)$, $v = k(ga, hgb)$, and $\theta = [ga/x, a/y, b/z]$. The nontrivial equivalence classes of the

congruence closure Π of E are $\{fx, fgy, gy\}$, and $\{hgz, gz\}$. Then, since $\theta(x) = \theta(gy) = ga$, we have $x \equiv_{\theta,\Pi} gy$ and $fx \equiv_{\theta,\Pi} fgy$. Thus, $\equiv_{\theta,\Pi}$ has two nontrivial equivalence classes $\{x, gy\}$ and $\{fx, fgy\}$. Assuming that we have a total simplification ordering on ground terms such that $a \prec b \prec f \prec g \prec h$ (for instance, a *recursive path ordering*, see Dershowitz [11]), we also have

$$y \preceq_{\theta,\Pi} z \preceq_{\theta,\Pi} x \preceq_{\theta,\Pi} gy_{\theta,\Pi} gz \preceq_{\theta,\Pi} fx \preceq_{\theta,\Pi} fgy \preceq_{\theta,\Pi} hgz.$$

The other pairs in $\preceq_{\theta,\Pi}$ are obtained by reflexivity and transitivity from $\equiv_{\theta,\Pi}$ and the above pairs.

This time, it is not obvious how to orient the equation $fx \doteq fgy$. This is because $\theta(fx) = \theta(fgy)$. One might think that this is a problem, but it can be overcome. Observe that since the ground equation $\theta(fx) \doteq \theta(fgy)$ is trivial, it does not help in any way in proving that $\theta(u)$ and $\theta(v)$ are congruent modulo $\theta(E)$. In [22], the problem was solved by factoring out the preorder $\preceq_{\theta,\Pi}$ by the equivalence relation $\equiv_{\theta,\Pi}$. It was also shown that as far as the completeness of the method is concerned, we can restrict our attention to partial orders rather than preorders. For the sake of simplicity, we present this solution, referring the reader to [22] for a more complete solution.

The key point is that it is always possible to choose an orientation of the equations which is compatible with $\preceq_{\theta,\Pi}$. For example, we can define the partial order $\preceq_{\mathscr{O}}$ on $\{x, y, z, fx, gy, gz, fgy, hgz\}$ such that, $gy \preceq_{\mathscr{O}} gz$, $gy \preceq_{\mathscr{O}} fgy$, $fgy \preceq_{\mathscr{O}} fx$, and $gz \preceq_{\mathscr{O}} hgz$ (other pairs in $\preceq_{\mathscr{O}}$ are obtained by transitivity and reflexivity). It is clear that $\preceq_{\mathscr{O}} \subseteq \preceq_{\theta,\Pi}$. With this orientation, the set E of Example 8.9 is equivalent to the following rigid reduced set of rewrite rules: $R = \{fx \to gy, fgy \to gy, hgz \to gz\}$.

The above discussion leads to the following definition that makes use of the fact noted before Definition 8.8.

Definition 8.10. Let $\preceq$ be a total simplification ordering on ground terms. Given a finite set S of terms and a partition Π on $ST(S)$, a partial order $\mathscr{O}$ on $ST(S)$ (also denoted as $\preceq_{\mathscr{O}}$) is an *order assignment for* Π iff the following properties hold:

(1) $\preceq_{\mathscr{O}}$ has the subterm property and is monotonic on $ST(S)$, that is, for all $u_1, \ldots, u_n, v_1, \ldots, v_n \in ST(S)$, if $u_i \preceq_{\mathscr{O}} v_i$ for $i = 1, \ldots, n$ and $f(u_1, \ldots, u_n)$ and $f(v_1, \ldots, v_n) \in ST(S)$, then $f(u_1, \ldots, u_n) \preceq_{\mathscr{O}} f(v_1, \ldots, v_n)$;

(2) The restriction of $\preceq_{\mathscr{O}}$ to ground terms agrees with $\preceq$ (on $ST(S)$), every nontrivial equivalence class C of Π has a least element, and $\preceq_{\mathscr{O}}$ is total on this set of least elements.

Given a finite set E of equations, if Π is the partition associated with the congruence closure of E, by an *order assignment for* E we mean an order assignment for Π.

The following lemma shows why order assignments can be chosen to be partial orders.

LEMMA 8.11. *Given a finite set S of terms and a partition Π on $ST(S)$, given any ground substitution θ such that $Var(\Pi) \subseteq D(\theta)$, there exists an order assignment $\preceq_{\mathscr{O}}$ for Π such that $\preceq_{\mathscr{O}} \subseteq \preceq_{\theta,\Pi}$ and $\preceq_{\mathscr{O}}$ is a total ordering.*

PROOF. For every nontrivial equivalence class C modulo $\equiv_{\theta,\Pi}$, we extend the simplification ordering $\prec$ as follows: Whenever such a class contains some

variable, say $C = \{x_1, \ldots, x_k, t_1, \ldots, t_m\}$ where $x_1, \ldots, x_k$ are variables, we extend $\prec$ to a relation $\prec'$ such that $x_1 \prec' x_2 \prec' \cdots \prec' x_k$ and $x_i \prec' t_j$, for all i, j, $1 \leq i \leq k$, $1 \leq j \leq m$. It is clear that $\preceq'$ is a partial ordering contained in $\preceq_{\theta,\Pi}$. Now, we define $\prec_{\mathscr{O}}$ recursively as follows: $u \prec_{\mathscr{O}} v$ iff either

(1) $\theta(u) \prec \theta(v)$, or
(2) $\theta(u) = \theta(v)$, and either

(2a) u is a variable and $u \prec' v$, or
(2b) $u = f(u_1, \ldots, u_n)$, $v = f(v_1, \ldots, v_n)$, and $\langle u_1, \ldots, u_n \rangle \prec_{\mathscr{O}}^{\text{lex}} \langle v_1, \ldots, v_n \rangle$, where $\prec_{\mathscr{O}}^{\text{lex}}$ is the lexicographic extension of $\prec_{\mathscr{O}}$.

We define $\preceq_{\mathscr{O}}$ as the reflexive closure of $\prec_{\mathscr{O}}$, and we claim that $\preceq_{\mathscr{O}}$ is a total ordering which is an order assignment contained in $\preceq_{\theta,\Pi}$. The only problem is in showing that $\preceq_{\mathscr{O}}$ is a total ordering, as the other conditions are then easily verified. To prove that $\preceq_{\mathscr{O}}$ is a total ordering, due to clause (1) of the definition of $\prec_{\mathscr{O}}$, it is enough to show that for any two distinct elements u, v in some nontrivial class C modulo $\equiv_{\theta,\Pi}$, either $u \preceq_{\mathscr{O}} v$ or $v \preceq_{\mathscr{O}} u$, but not both. Note that the set of classes modulo $\equiv_{\theta,\Pi}$ is totally ordered: $C \ll C'$ iff $\theta(C) \prec \theta(C')$, where $\theta(C)$ denotes the common value of all terms $\theta(t)$ where $t \in C$. We proceed by induction on this well-ordering of the classes. Clearly, the least class contains some variable and at most one constant. But then, it is already totally ordered by $\prec'$. Given any other nontrivial class C, if u and v are both variables, we already know by (2a) that either $u \prec' v$ or $v \prec' u$, but not both. If u is a variable and v is not, by (2a) we can only have $u \prec' v$. If both u and v are not variables, then they must be of the form $u = f(u_1, \ldots, u_n)$ and $v = f(v_1, \ldots, v_n)$, since C is unified by θ. Since $u \neq v$, there is a least i such that $u_i \neq v_i$, and since θ unifies u and v, θ unifies u_i and v_i. But then, because $\prec$ has the subterm property, u_i, v_i belong to some class C_i such that $C_i \ll C$. Therefore, either $u_i \preceq_{\mathscr{O}} v_i$ or $v_i \preceq_{\mathscr{O}} u_i$, but not both, and thus by (2b), either $u \preceq_{\mathscr{O}} v$ or $v \preceq_{\mathscr{O}} u$, but not both. □

In view of Lemma 8.11, the following definition is justified.

Definition 8.12. Given a finite set of terms S, an order assignment $\preceq_{\mathscr{O}}$ for a partition Π on $ST(S)$ is *realized* by a ground substitution θ such that $Var(\Pi) \subseteq D(\theta)$ iff $\preceq_{\mathscr{O}} \subseteq \preceq_{\theta,\Pi}$.

Given two order assignments $\mathscr{O}$ on a partition Π for $ST(S)$ and $\mathscr{O}'$ on a partition Π' for $ST(S')$, we say that $\mathscr{O}$ and $\mathscr{O}'$ are *compatible* iff they coincide on $ST(S) \cap ST(S')$.

Example 8.13. Let $E = \{fx \doteq fgy, fgy \doteq gy, hgz \doteq gz\}$, as in Example 8.9. The nontrivial equivalence classes of the congruence closure Π of E are $\{fx, fgy, gy\}$, and $\{hgz, gz\}$.

Let $\mathscr{O}$ be the partial order on $\{x, y, z, fx, gy, gz, fgy, hgz\}$ such that $gy \preceq_{\mathscr{O}} gz$, $gy \preceq_{\mathscr{O}} fgy$, $fgy \preceq_{\mathscr{O}} fx$, and $gz \preceq_{\mathscr{O}} hgz$ (other pairs in $\preceq_{\mathscr{O}}$ are obtained by transitivity and reflexivity). It is immediately verified that $\mathscr{O}$ is an order assignment realized by $\theta = [ga/x, a/y, b/z]$, since $\preceq_{\mathscr{O}} \subseteq \preceq_{\theta,\Pi}$.

The next example arises from the problem of proving that every monoid such that $x \cdot x = 1$ (for all x) is commutative.

Example 8.14. Let $\mathscr{E}$ be the set of equations

$$\begin{aligned}
\mathscr{E} = \{ & u_1 \cdot 1 \doteq u_1 \\
& w_1 \cdot w_1 \doteq 1 \\
& x_1 \cdot (y_1 \cdot z_1) \doteq (x_1 \cdot y_1) \cdot z_1 \\
& x_2 \cdot (y_2 \cdot z_2) \doteq (x_2 \cdot y_2) \cdot z_2 \\
& w_2 \cdot w_2 \doteq 1 \\
& 1 \cdot v_1 \doteq v_1 \\
& x_3 \cdot (y_3 \cdot z_3) \doteq (x_3 \cdot y_3) \cdot z_3 \\
& x_4 \cdot (y_4 \cdot z_4) \doteq (x_4 \cdot y_4) \cdot z_4 \\
& w_3 \cdot w_3 \doteq 1 \\
& eq(a \cdot b, b \cdot a) \doteq F \\
& eq(z, z) \doteq T \}.
\end{aligned}$$

The nontrivial equivalence classes of the congruence closure Π of $\mathscr{E}$ are:

$$\begin{aligned}
& \{T, eq(z, z)\}, \\
& \{F, eq(a \cdot b, b \cdot a)\}, \\
& \{1, w_2 \cdot w_2, w_3 \cdot w_3, w_1 \cdot w_1\}, \\
& \{u_1, u_1 \cdot 1\}, \\
& \{v_1, 1 \cdot v_1\}, \\
& \{x_2 \cdot (y_2 \cdot z_2), (x_2 \cdot y_2) \cdot z_2\}, \\
& \{x_4 \cdot (y_4 \cdot z_4), (x_4 \cdot y_4) \cdot z_4\}, \\
& \{x_3 \cdot (y_3 \cdot z_3), (x_3 \cdot y_3) \cdot z_3\}, \\
& \{x_1 \cdot (y_1 \cdot z_1), (x_1 \cdot y_1) \cdot z_1\}.
\end{aligned}$$

We define the order assignment $\mathscr{O}$ on Π by the order in which the elements in each class of Π are listed, and for the least elements in these classes, the order in which the classes are listed. All other pairs in $\preceq_{\mathscr{O}}$ are determined by reflexivity and transitivity. It is easily seen that there is a total simplification ordering on ground terms such that $1 \prec a \prec b \prec \cdot$, and one can verify that $\preceq_{\mathscr{O}}$ is an order assignment, and that $\preceq_{\mathscr{O}}$ is realized by the substitution

$$\begin{aligned}
\theta = [& a/u_1, a/x_1, a/x_2, a/y_2, a/w_2, a/x_4, \\
& b/z_2, b/v_1, b/x_3, b/z_3, b/y_4, b/z_4, b/w_3, \\
& a \cdot b/w_1, a \cdot b/y_1, a \cdot b/z_1, a \cdot b/y_3, a \cdot b/z].
\end{aligned}$$

We can now modify the procedure of Definition 8.4 in order to accommodate variables.

Definition 8.15 (*Reduction procedure R*). Let $\prec$ be a total simplification ordering on ground terms. Let $\mathscr{E} = \mathscr{E}_\Sigma \cup \{eq(u, v) \doteq F, eq(z, z) \doteq T\}$ be a finite set of equations, where $\mathscr{E}_\Sigma$ is a set of equations over $T_\Sigma(X)$, and $u, v \in T_\Sigma(X)$. Given any order assignment $\mathscr{O}$ on $\mathscr{E}$, the procedure R returns a

rigid reduced rewrite system $R(\mathscr{E}, \mathscr{O})$. To form the system $R(\mathscr{E}, \mathscr{O})$, since $\preceq_{\mathscr{O}}$ is a simplification ordering such that every nontrivial equivalence class of Π has a least element and $\preceq_{\mathscr{O}}$ is total on this set of least elements, we apply to $\mathscr{E}$ and Π the procedure described in Definition 8.4, except that at the end of every round, it may be necessary to extend $\mathscr{O}$ since new terms may arise due to simplification. If at every round an extension of $\mathscr{O}$ can be found so that the next step can be performed, R succeeds and returns a rigid reduced rewrite system denoted as $R(\mathscr{E}, \mathscr{O})$. Otherwise, R returns failure.

It is useful to remark that since the reduction procedure deals with sets of equations of the form $\mathscr{E} = \mathscr{E}_\Sigma \cup \{eq(u, v) \doteq F, eq(z, z) \doteq T\}$, in the congruence closure Π of $\mathscr{E}$, the classes of T and F are always $\{eq(u, v), F\}$ and $\{eq(z,z), T\}$. From the way we have extended $\prec$ to take care of T, F, and terms involving eq, it will be shown as a corollary of Theorem 10.2 that there is no loss of generality in choosing order assignments such that $T \preceq_{\mathscr{O}} F \preceq_{\mathscr{O}} s \preceq_{\mathscr{O}} eq(u, v)$ for all $s, u, v \in T_\Sigma(X)$. We can show the following crucial result.

LEMMA 8.16. *Let $\mathscr{E} = \mathscr{E}_\Sigma \cup \{eq(u, v) \doteq F, eq(z, z) \doteq T\}$ be a finite set of equations, where $\mathscr{E}_\Sigma$ is a set of equations over $T_\Sigma(X)$, $u, v \in T_\Sigma(X)$, and $\prec$ a total simplification ordering on ground terms. Given an order assignment $\mathscr{O}$ on $\mathscr{E}$, if R does not fail, then $R(\mathscr{E}, \mathscr{O})$ is rigid equivalent to $\mathscr{E}$.*

We are now ready to define a procedure for finding rigid E-unifiers.

9. *A Method for Finding Complete Sets of Rigid E-Unifiers*

This method uses the reduction procedure of Section 8 and a single transformation on certain systems defined next. First, the following definition is needed.

Definition 9.1. Given a set E of equations and some equation $l \doteq r$, the set of equations obtained from E by deleting $l \doteq r$ and $r \doteq l$ from E is denoted by $(E - \{l \doteq r\})^\dagger$. Formally, we let $(E - \{l \doteq r\})^\dagger = \{u \doteq v \mid u \doteq v \in E, u \doteq v \neq l \doteq r$, and $u \doteq v \neq r \doteq l\}$.

Definition 9.2. Let $\prec$ be a total simplification ordering on ground terms. We shall be considering finite sets of equations of the form $\mathscr{E} = \mathscr{E}_\Sigma \cup \{eq(u, v) \doteq F, eq(z, z) \doteq T\}$, where $\mathscr{E}_\Sigma$ is a set of equations over $T_\Sigma(X)$, and $u, v \in T_\Sigma(X)$. We define a transformation on systems of the form $\langle \mathscr{S}, \mathscr{E}, \mathscr{O} \rangle$, where $\mathscr{S}$ is a term system, $\mathscr{E}$ a set of equations as above and $\mathscr{O}$ an order assignment:

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow \langle \mathscr{S}_1, \mathscr{E}_1, \mathscr{O}_1 \rangle,$$

where $l_1 \doteq r_1, l_2 \doteq r_2 \in \mathscr{E}_0 \cup \mathscr{E}_0^{-1}$, either l_1/β is *not* a variable or $l_2 \doteq r_2$ is degenerate, $l_1/\beta \neq l_2, TU(l_1/\beta, l_2)$ represents a mgu of l_1/β and l_2 in triangular form,[17] $\sigma = [t_1/x_1, \ldots, t_p/x_p]$ where $TU(l_1/\beta, l_2) = \{\langle x_1, t_1 \rangle, \ldots, \langle x_p, t_p \rangle\}$,

$$\mathscr{E}_1' = \sigma\left((\mathscr{E}_0 - \{l_1 \doteq r_1\})^\dagger \cup \{l_1[\beta \leftarrow r_2] \doteq r_1\}\right),$$

$\mathscr{O}_1$ is an order assignment on $\mathscr{E}_1'$ compatible with $\mathscr{O}_0$, $\mathscr{S}_1 = \mathscr{S}_0 \cup TU(l_1/\beta, l_2)$, and $\mathscr{E}_1 = R(\mathscr{E}_1', \mathscr{O}_1)$.

[17] Note that we are requiring that l_1/β and l_2 have a *nontrivial* unifier. The triangular form of mgus important for the NP-completeness of this method.

Observe that $\sigma(l_1[\beta \leftarrow r_2] \doteq r_1)$ looks like a critical pair of equations in $\mathscr{E}_0 \cup \mathscr{E}_0^{-1}$, but it is not. This is because a critical pair is formed by applying the mgu of l_1/β and l_2 to $l_1[\beta \leftarrow r_2] \doteq r_1$, but $[t_1/x_1, \ldots, t_p/x_p]$ is usually not a mgu of l_1/β and l_2. It is the composition $[t_1/x_1]; \cdots; [t_p/x_p]$ that is a mgu of l_1/β and l_2. The reason for not applying the mgu is that by repeated applications of this step, exponential size terms could be formed, and it would not be clear that the decision procedure is in NP. We have chosen an approach of "lazy" (or delayed) unification. Also note that we use the rigid reduced system $R(\mathscr{E}_1', \mathscr{O}_1)$ rather than $\mathscr{E}_1'$, and so, a transformation step is defined only if R does not fail. The method then is the following:

Definition 9.3 (*Method*). Let $E_{u,v} = E \cup \{eq(u, v) \doteq F, eq(z, z) \doteq T\}$, $\mathscr{O}_0$ an order assignment on $E_{u,v}$, $\mathscr{S}_0 = \emptyset$, $\mathscr{E}_0 = R(E_{u,v}, \mathscr{O}_0)$, m the total number of variables in $\mathscr{E}_0$, and $V = Var(E) \cup Var(u, v)$. For any sequence

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle$$

consisting of at most m transformation steps, if $\mathscr{S}_k$ is unifiable and $k \leq m$ is the first integer in the sequence such that $F \doteq T \in \mathscr{E}_k$, return the substitution $\theta_{\mathscr{S}_k}|_V$, where $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ (over $T_\Sigma(X)$).

Example 9.4. Let E be the set of equations $E = \{fa \doteq a, ggx \doteq fa\}$, and $\langle u, v \rangle = \langle gggx, x \rangle$. We have

$$E_{u,v} = \{fa \doteq a, ggx \doteq fa, eq(gggx, x) \doteq F, eq(z, z) \doteq T\}.$$

The congruence closure Π of $E_{u,v}$ has three nontrivial classes $\{a, fa, ggx\}$, $\{eq(gggx, x), F\}$, and $\{eq(z, z), T\}$. Let $\mathscr{O}_0$ be the order assignment on $E_{u,v}$ such that

$$T \prec_{\mathscr{O}_0} eq(gggx, x),$$
$$F \prec_{\mathscr{O}_0} eq(z, z),$$
$$a \prec_{\mathscr{O}_0} fa \prec_{\mathscr{O}_0} ggx,$$

the least elements of classes being ordered in the order of listing of the classes. We have $\mathscr{S}_0 = \emptyset$, and the reduced system $\mathscr{E}_0 = R(E_{u,v}, \mathscr{O}_0)$ is

$$\mathscr{E}_0 = \{fa \doteq a, ggx \doteq a, eq(ga, x) \doteq F, eq(z, z) \doteq T\}.$$

Note that there is an overlap between $eq(ga, x) \doteq F$ and $eq(z, z) \doteq T$ at address ϵ in $eq(ga, x)$, and we obtain the triangular system $\{\langle x, ga \rangle, \langle z, ga \rangle\}$ and the new equation $F \doteq T$. Thus, we have

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow \langle \mathscr{S}_1, \mathscr{E}_1, \mathscr{O}_1 \rangle,$$

where $\mathscr{S}_1 = \{\langle x, ga \rangle, \langle z, ga \rangle\}$

$$\mathscr{E}_1' = \{fa \doteq a, ggga \doteq a, eq(ga, ga) \doteq F, F \doteq T\},$$

and $\mathscr{O}_1$ is the restriction of $\mathscr{O}_0$ to the subterms in $\mathscr{E}_1'$. After reducing $\mathscr{E}_1'$, we have

$$\mathscr{E}_1 = \{fa \doteq a, ggga \doteq a, eq(ga, ga) \doteq T, F \doteq T\}.$$

Since $F \doteq T \in \mathscr{E}_1$ and $\mathscr{S}_1$ is unifiable, the restriction $[ga/x]$ of the mgu $[ga/x, ga/z]$ of $\mathscr{S}_1$ to $Var(E) \cup Var(u, v) = \{x\}$ is a rigid E-unifier of $gggx$ and x.

10. *Soundness, Completeness, and Decidability of the Rigid E-Unification Method*

The main properties of the method are given in this section.

THEOREM 10.1 (SOUNDNESS). *Let E be a set of equations over $T_\Sigma(X)$, u, v two terms in $T_\Sigma(X)$, $E_{u,v} = E \cup \{eq(z, z) \doteq T, eq(u, v) \doteq F\}$, $\mathscr{O}_0$ an order assignment on $E_{u,v}$, $\mathscr{S}_0 = \emptyset$, $\mathscr{E}_0 = R(E_{u,v}, \mathscr{O}_0)$, m the total number of variables in $\mathscr{E}_0$, and $V = Var(E) \cup Var(u, v)$. If*

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle,$$

where $\mathscr{S}_k$ is unifiable, $F \doteq T \in \mathscr{E}_k$ and $F \doteq T \notin \mathscr{E}_i$ for all i, $0 \le i < k \le m$, then $\theta_{\mathscr{S}_k}|_V$ is a rigid E-unifier of u and v, where $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ (over $T_\Sigma(X)$).

The proof of Theorem 10.1 does not use the fact that the systems $R(\mathscr{E}_i', \mathscr{O}_i)$ are rigid reduced, but only the fact that $\theta'(\mathscr{E}_i)$ and $\theta'(R(\mathscr{E}_i', \mathscr{O}_i))$ are rigid equivalent. However, the fact that the systems $R(\mathscr{E}_i', \mathscr{O}_i)$ are rigid reduced plays a crucial role in the proof of the completeness theorem. The $\mathscr{O}_i$'s are only needed for the completeness of the method, and to make sure that the reduction procedure terminates. We now turn to the completeness part. The main technique is roughly the removal of peaks by the use of critical pairs (Bachmair [4], Bachmair et al. [5, 6]).

THEOREM 10.2 (COMPLETENESS). *Let E be a set of equations over $T_\Sigma(X)$ and u, v two terms in $T_\Sigma(X)$. If θ is any rigid E-unifier of u and v, then there is an order assignment $\mathscr{O}_0$ on $E_{u,v}$, and letting $\mathscr{S}_0 = \emptyset$, $\mathscr{E}_0 = R(E_{u,v}, \mathscr{O}_0)$, m the number of variables in $R(\mathscr{E}_{u,v}, \mathscr{O}_0)$, and $V = Var(E) \cup Var(u, v)$, there is a sequence of transformations*

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle,$$

where $k \le m$, $\mathscr{S}_k$ is unifiable, $F \doteq T \in \mathscr{E}_k$, $F \doteq T \notin \mathscr{E}_i$ for all i, $0 \le i < k$, and $\theta_{\mathscr{S}_k}|_V \le_E \theta[V]$, where $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ over $T_\Sigma(X)$. Furthermore, $\theta_{\mathscr{S}_k}|_V$ is a rigid E-unifier of u and v.

COROLLARY 10.3. *If θ' is the mgu produced by a sequence of steps as in the soundness theorem, there is a ground substitution θ_1 such that $V \subseteq D(\theta_1)$ and a sequence of steps*

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle,$$

such that $\theta_1 \sqsubseteq_E \widehat{\theta'}$, θ_1 is a unifier of $\mathscr{S}_k$, and θ_1 realizes all the $\mathscr{O}_i$'s in the above sequence. In particular, the method is still complete if we restrict ourselves to order assignments $\mathscr{O}$ such that $T \preceq_{\mathscr{O}} F \preceq_{\mathscr{O}} s \preceq_{\mathscr{O}} eq(u, v)$ for all $s, u, v \in T_\Sigma(X)$.

Theorem 10.2 also shows that rigid E-unification is decidable.

COROLLARY 10.4. *Rigid E-unification is decidable.*

Combining the results of Theorem 10.1 and 10.2, we also obtain the fact that for any E, u, v, there is always a finite complete set of rigid E-unifiers.

THEOREM 10.5. *Let E be a set of equations over $T_\Sigma(X)$, u, v two terms in $T_\Sigma(X)$, m the number of variables in $E \cup \{u, v\}$, and $V = Var(E) \cup Var(u, v)$. There is a finite complete set of rigid E-unifiers for u and v given by the set*

$$\{\theta_{\mathscr{S}_k} |_V \mid \langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle, k \leq m\},$$

for any order assignment $\mathscr{O}_0$ on $E_{u,v}$, with $\mathscr{S}_0 = \emptyset$, $\mathscr{E}_0 = R(E_{u,v}, \mathscr{O}_0)$, and where $\mathscr{S}_k$ is unifiable, $F \doteq T \in \mathscr{E}_k$, $F \doteq T \notin \mathscr{E}_i$ for all i, $0 \leq i < k$, and $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ over $T_\Sigma(X)$.

11. *NP-Completeness of Rigid E-Unification*

First, recall that rigid E-unification is NP-hard. This holds even for ground sets of equations, as shown by Kozen [30, 31]. Using an idea of Kozen [30], we show that rigid E-unification is NP-hard even when all equations in E are regular, all ground except one, and u and v are ground.

Definition 11.1. An equation $(l \doteq r)$ is *regular* iff $Var(l) = Var(r)$.

THEOREM 11.2. *Rigid E-unification is NP-hard when all equations in E are regular, all ground except one, and u and v are ground.*

PROOF. The satisfiability problem is reduced to rigid E-unification as follows. Let the set of function symbols consist of $\wedge$, $\vee$, $\neg$, and the constants $\top$ and $\bot$. Write down the set E_{bool} of 10 ground equations corresponding to the truth tables for $\wedge$, $\vee$, $\neg$. Given any clause A, if $Var(A) = \{x_1, \ldots, x_n\}$, let

$$B_A = (x_1 \wedge x_2 \wedge \cdots \wedge x_n \wedge \bot).$$

Finally, let $E_A = E_{bool} \cup \{A \doteq B_A\}$, $u = \top$ and $v = \bot$. Clearly, $A \doteq B_A$ is regular, and it is easy to see that a substitution σ such that $\top$ and $\bot$ are congruent modulo $\sigma(E_A)$ exists iff A is satisfiable, since B_A is false for every truth assignment. Hence, satisfiability is reduced to rigid E-unification. □

We now show that rigid E-unification is in NP.

THEOREM 11.3. *Rigid E-unification is NP-complete.*

PROOF. We already know that rigid E-unification is NP-hard. By Corollary 10.4, the problem is decidable. It remains to show that it is in NP. From Corollary 10.4, u and v have some rigid E-unifier iff there is some sequence of transformations

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle$$

of at most $k \leq m$ steps where m is the number of variables in $\mathscr{E}_0$, and such that $\mathscr{S}_k$ is unifiable (over $T_\Sigma(X)$), $F \doteq T \in \mathscr{E}_k$ and $F \doteq T \notin \mathscr{E}_i$ for all i, $0 \leq i < k$. We need to verify that it is possible to check these conditions in polynomial time. First, observe that a triangular form can be computed in polynomial time, applying the substitutions associated with triangular forms can also be done in polynomial time, and checking that a preorder is an order assignment can be done in polynomial time. Finally, we need to show that the total cost of producing reduced systems is polynomial. This is a crucial point that had been overlooked in a previous version of this paper, and we thank Leo

Bachmair for pointing out this subtlety to us. We use two facts that have to do with implementing the steps of the algorithm using term DAGs.

(1) We have already noted (see Theorem 8.7) that the size of the term DAG associated with a reduced system equivalent to an input set of equations is no greater than the size of the input term DAG, the number of rules no greater than the number of input equations, and that the reduction procedure runs in $O((m + n + p)^3)$, where (m, n) is the size of the input term DAG and p the number of equations in E.
(2) The term DAG associated with the system $\mathscr{E}'_{i+1}$ obtained from $\mathscr{E}_i$ by a transformation step can be obtained from the term DAG associated with $\mathscr{E}_i$ by moving pointers, and if (m', n') and (m, n) are the sizes of the term DAGs of the systems $\mathscr{E}'_{i+1}$ and $\mathscr{E}_i$, respectively, and p' and p the numbers of equations in these systems, then $m' \leq m$, $n' \leq n$, and $p' \leq p$.

The reason why (2) holds is that the terms occurring in the triangular form of the substitution σ associated with the transformation step all belong to the term DAG associated with $\mathscr{E}_i$. For instance, this is easily seen if one uses Paterson and Wegman's method [37]. Now, forming $l_1[\beta \leftarrow r_2]$ only involves pointer redirection, and so does the application of σ. Thus, the size of the resulting term DAG cannot increase. By the definition of the transformations, it is also obvious that $p' \leq p$.

Because the number of steps is at most the number of variables in $\mathscr{E}_0$, the total cost of producing reduced systems is indeed polynomial in the size of the input.

It is interesting to note the analogy of this part of our proof with Kozen's proof that his method is in NP [31]. Both use the term DAG representation in a crucial way. In this way, we avoid the potential exponential explosion that can take place during reductions if identical subterms are not shared. □

If E is a set of ground equations, the $\mathscr{O}_i$'s are useless and the reduction procedure R needs only be applied once at the beginning of E. Thus, Theorem 11.3 provides another proof of a result first established by Kozen [30, 31].

12. *Applications of Rigid E-Unification to Equational Matings*

The method developed for one set of equations and one pair can be easily generalized to tackle problem (1). In fact, an algorithm to decide whether a family of mated sets is an equational (pre)mating is obtained. The method of Definition 9.3 can be generalized to pairs $\langle \vec{E}, S \rangle$ (as defined in problem 1 in the introduction) by considering triples $\langle \mathscr{S}, \mathscr{E}, \mathscr{O} \rangle$, where $\mathscr{S}$ is a term system, and $\mathscr{E}$ is an n-tuple of sets of equations. The definition of a rigid $\vec{E}$-unifier of a set of pairs is generalized as follows:

Definition 12.1. Let $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ be a family of n sets of equations (over $T_\Sigma(X)$) and $S = \{\langle u_i, v_i \rangle \mid 1 \leq i \leq n\}$ a set of n pairs of terms (over $T_\Sigma(X)$). A substitution θ (over $T_\Sigma(X)$) is a *rigid $\vec{E}$-unifier of S* iff

$$\theta(u_i) \doteq_{\theta(E_i)} \theta(v_i)$$

for every i, $1 \leq i \leq n$. A pair $\langle \vec{E}, S \rangle$ such that S has some rigid $\vec{E}$-unifier is called an *equational premating*.

The suitable generalization of the preorder $\leq_{\vec{E}}$ to a family $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ of n sets of equations turns out to be the following:

Definition 12.2. Given a family $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ of n sets of equations, for any (finite) set of variables V, for any two substitutions σ and θ, $\sigma \leq_{\vec{E}} \theta$ iff there is some η such that $\sigma; \eta \sqsubseteq_{E_i} \theta[V]$ for every i, $1 \leq i \leq n$.

Note that this condition is stronger than the condition $\sigma \leq_{E_i} \theta[V]$ for every i, $1 \leq i \leq n$, because with this second condition we only know that there are substitutions $\eta_1, \ldots, \eta_n$ such that $\sigma; \eta_i \sqsubseteq_{E_i} \theta[V]$ for every i, $1 \leq i \leq n$. In Definition 12.2, it is required that $\eta_1 = \cdots = \eta_n$. It is straightforward to verify that the generalization of Theorem 10.2 holds with the stronger Definition 12.2.

Complete sets of rigid $\vec{E}$-unifiers for S are defined as follows:

Definition 12.3. Let $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ and $S = \{\langle u_i, v_i \rangle \mid 1 \leq i \leq n\}$ as in Definition 12.1, and let $V = Var(\vec{E}) \cup Var(S)$. A set U of substitutions is a *complete set of rigid $\vec{E}$-unifiers for S* iff: For every $\sigma \in U$,

(i) $D(\sigma) \subseteq V$ and $D(\sigma) \cap I(\sigma) = \varnothing$ (idempotence),
(ii) σ is a rigid $\vec{E}$-unifier of S,
(iii) For every rigid $\vec{E}$-unifier θ of S, there is some $\sigma \in U$ such that $\sigma \leq_{\vec{E}} \theta[V]$.

Minimal rigid $\vec{E}$-unifiers also exist and are defined as follows:

Definition 12.4. Let $\vec{E}$ be a family of sets of equations and S a term system as in Definition 12.1. For any ground rigid $\vec{E}$-unifier θ of S, let

$$S_{\vec{E},S,\theta} = \left\{\rho \mid D(\rho) = D(\theta), \rho(u_i) \doteq_{\rho(E_i)} \rho(v_i), \rho \sqsubseteq_{E_i} \theta, 1 \leq i \leq n, \text{ and } \rho \text{ ground}\right\}.$$

Since $\lessdot\!\lessdot$ is total and well-founded on ground substitutions with domain $D(\theta)$, the set $S_{\vec{E},S,\theta}$ contains some least element σ (with respect to $\lessdot\!\lessdot$).

It is easy to see that Lemma 7.4 can be generalized as follows:

LEMMA 12.5. *Let $\vec{E}$ be a family of sets of equations and S a term system as in Definition* 12.1. *For any ground rigid $\vec{E}$-unifier θ of S, if σ is the least element of the set $S_{\vec{E},S,\theta}$ of Definition* 12.4, *then the following properties hold*:

(1) $\sigma \sqsubseteq_{E_i} \theta$ for every i, $1 \leq i \leq n$,
(2) every term of the form $\sigma(x)$ is irreducible by every oriented instance $\sigma(l) \to \sigma(r)$ of a nondegenerate equation $l \doteq r \in \vec{E} \cup \vec{E}^{-1}$, and
(3) every proper subterm of a term of the form $\sigma(x)$ is irreducible by every oriented instance $\sigma(l) \to \sigma(r)$ of a degenerate equation $l \doteq r \in \vec{E} \cup \vec{E}^{-1}$.

Lemma 8.3 is easily generalized as follows: We let $eq_1, \ldots, eq_n$ be n new distinct binary function symbols not in Σ (and distinct from T and F).

LEMMA 12.6. *Let $\vec{E}$ be a family of sets of equations and S a term system as in Definition* 12.1. *A substitution θ over $T_\Sigma(X)$ is a rigid $\vec{E}$-unifier of S iff there is some substitution θ' over $T_\Sigma(X)$ such that $\theta = \theta' \mid_{D(\theta') - \{z_1, \ldots, z_n\}}$ and $T \doteq_{\theta'(E')} F$ for every i, $1 \leq i \leq n$, where $E' = E_i \cup \{eq_i(u_i, v_i) \doteq F, eq_i(z_i, z_i) \doteq T\}$, and $\{z_1, \ldots, z_n\}$ is a set of new variables not in $Var(\vec{E}) \cup Var(S)$.*

The total simplification ordering $\prec$ is extended to the set

$$T_\Sigma \cup \{T, F\} \cup \bigcup_{i=1}^{i=n} \{eq_i(u, v) \mid u, v \in T_\Sigma\}$$

as follows:

For any terms $s, t, u, v \in T_\Sigma$,

(a) $T \prec F \prec u \prec eq_i(s, t)$;
(b) $eq_i(s, t) \prec eq_l(u, v)$ iff $\{s, t\} \prec_{lex} \{u, v\}$, where $\prec_{lex}$ is the lexicographic extension of $\prec$ to pairs;
(c) $eq_i(s, t) \prec eq_j(u, v)$ iff $1 \leq i < j \leq n$.

Clearly, this extension of $\prec$ is a total simplification ordering. We define a transformation on systems as follows: We shall be considering n-tuples $\mathscr{E} = \langle \mathscr{E}^1, \ldots, \mathscr{E}^n \rangle$ of finite sets of equations of the form $\mathscr{E}^i = \mathscr{E}^i_\Sigma \cup \{eq_i(u, v) \doteq F, eq_i(z_i, z_i) \doteq T\}$, where $\mathscr{E}_\Sigma$ is a set of equations over $T_\Sigma(X)$ and $u, v \in T_\Sigma(X)$. We define a transformation on systems of the form $\langle \mathscr{S}, \mathscr{E}, \mathscr{O} \rangle$, where $\mathscr{S}$ is a term system, $\mathscr{E}$ an n-tuple of sets of equations as above, and $\mathscr{O}$ an order assignment:

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow \langle \mathscr{S}_1, \mathscr{E}_1, \mathscr{O}_1 \rangle,$$

where $l_1 \doteq r_1$, $l_2 \doteq r_2 \in \mathscr{E}^i_0 \cup (\mathscr{E}^i_0)^{-1}$ for some i, $1 \leq i \leq n$, either l_1/β is *not* a variable or $l_2 \doteq r_2$ is degenerate, $l_1/\beta \neq l_2$, $TU(l_1/\beta, l_2)$ represents a mgu of l_1/β and l_2 in triangular form, $\sigma = [t_1/x_1, \ldots, t_p/x_p]$ where $TU(l_1/\beta, l_2) = \{\langle x_1, t_1 \rangle, \ldots, \langle x_p, t_p \rangle\}$,

$$\mathscr{E}'^i = \sigma\left(\left(\mathscr{E}^i_0 - \{l_1 \doteq r_1\}\right)^\dagger \cup \{l_1[\beta \leftarrow r_2] \doteq r_1\}\right)$$

$$\text{and} \qquad \mathscr{E}'^j_1 = \sigma\left(\mathscr{E}^j_0\right) \qquad \text{for every} \quad j \neq i,$$

$\mathscr{O}_1$ is an order assignment on $\mathscr{E}'_1$ compatible with $\mathscr{O}_0$, $\mathscr{S}_1 = \mathscr{S}_0 \cup TU(l_1/\beta, l_2)$, and $\mathscr{E}_1 = \langle \mathscr{E}^1_1, \ldots, \mathscr{E}^n_1 \rangle$, where $\mathscr{E}^j_1 = R(\mathscr{E}'^j_1, \mathscr{O}_1)$ for all j, $1 \leq j \leq n$.

The method for finding rigid $\vec{E}$-unifiers of S is the following:

Definition 12.7 (*Method*). Let $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ and $S = \{\langle u_i, v_i \rangle \mid 1 \leq i \leq n\}$ as in Definition 12.1, let $E^i = E_i \cup \{eq_i(u_i, v_i) \doteq F, eq_i(z_i, z_i) \doteq T\}$ for every i, $1 \leq i \leq n$, $\mathscr{O}_0$ an order assignment on $\langle E^1, \ldots, E^n \rangle$, $\mathscr{S}_0 = \emptyset$, $\mathscr{E}^i_0 = R(E^i, \mathscr{O}_0)$ for every i, $1 \leq i \leq n$, $\mathscr{E}_0 = \langle \mathscr{E}^1_0, \ldots, \mathscr{E}^n_0 \rangle$, m the total number of variables in $\mathscr{E}_0$, and $V = Var(\vec{E}) \cup Var(S)$. For any sequence

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle$$

consisting of at most m transformation steps, if $\mathscr{S}_k$ is unifiable and $k \leq m$ is the first integer in the sequence such that $F \doteq T \in \mathscr{E}^i_k$ for every i, $1 \leq i \leq n$, return the substitution $\theta_{\mathscr{S}_k}|_V$, where $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ (over $T_\Sigma(X)$).

The proofs of Theorem 10.1 and Theorem 10.2 can be easily adapted to prove that the finite set of all substitutions returned by the method of Definition 12.7 forms a complete set of rigid $\vec{E}$-unifiers for $\mathscr{S}$. In particular, the method provides a decision procedure for deciding whether a family of mated sets is an equational premating that is in NP.

THEOREM 12.8 (SOUNDNESS). *Let* $\vec{E} = \{E_i \mid 1 \leq i \leq n\}$ *and* $S = \{\langle u_i, v_i \rangle \mid 1 \leq i \leq n\}$ *as in Definition* 12.1, *let* $E^i = E_i \cup \{eq_i(u_i, v_i) \doteq F, eq_i(z_i, z_i) \doteq T\}$ *for every* i, $1 \leq i \leq n$, $\mathscr{O}_0$ *an order assignment on* $\langle E^1, \ldots, E^n \rangle$, $\mathscr{S}_0 = \emptyset$, $\mathscr{E}^i_0 =$

$R(E^i, \mathscr{O}_0)$ for every i, $1 \le i \le n$, $\mathscr{E}_0 = \langle \mathscr{E}_0^1, \ldots, \mathscr{E}_0^n \rangle$, m the total number of variables in $\mathscr{E}_0$, and $V = Var(\vec{E}) \cup Var(S)$. If

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle,$$

where $\mathscr{S}_k$ is unifiable, $F \doteq T \in \mathscr{E}_k^j$ and $F \doteq T \notin \mathscr{E}_i^j$ for all i and j, $0 \le i < k \le m$, $1 \le j \le n$, then $\theta_{\mathscr{S}_k}|_V$ is a rigid $\vec{E}$-unifier of S, where $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ (over $T_\Sigma(X)$).

THEOREM 12.9 (COMPLETENESS). *Let $\vec{E} = \{E_i \mid 1 \le i \le n\}$ and $S = \{\langle u_i, v_i \rangle \mid 1 \le i \le n\}$ as in Definition 12.1, and let $E^i = E_i \cup \{eq_i(u_i, v_i) \doteq F, eq_i(z_i, z_i) \doteq T\}$ for every i, $1 \le i \le n$. If θ is any rigid $\vec{E}$-unifier of S, then there is an order assignment $\mathscr{O}_0$ on $\langle E^1, \ldots, E^n \rangle$, and letting $\mathscr{S}_0 = \emptyset$, $\mathscr{E}_0^i = R(E^i, \mathscr{O}_0)$ for every i, $1 \le i \le n$, $\mathscr{E}_0 = \langle \mathscr{E}_0^1, \ldots, \mathscr{E}_0^n \rangle$, m the total number of variables in $\mathscr{E}_0$, and $V = Var(\vec{E}) \cup Var(S)$, there is a sequence of transformations*

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle,$$

where $k \le m$, $\mathscr{S}_k$ is unifiable, $F \doteq T \in \mathscr{E}_k^j$, $F \doteq T \notin \mathscr{E}_i^j$ for all i and j, $0 \le i < k$, $1 \le j \le n$, and $\theta_{\mathscr{S}_k}|_V \le_{\vec{E}} \theta[V]$, where $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ over $T_\Sigma(X)$. Furthermore, $\theta_{\mathscr{S}_k}|_V$ is a rigid $\vec{E}$-unifier of S.

Actually, Theorem 12.9 can be sharpened. Examination of the induction proof reveals that for any rigid $\vec{E}$-unifier θ of S, a rigid $\vec{E}$-unifier more general than θ can be found, even if the transformations are applied in a certain order.

Definition 12.10. We say that a derivation

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_m, \mathscr{E}_m, \mathscr{O}_m \rangle$$

is an *lr-derivation* iff for every subderivation

$$\langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^* \langle \mathscr{S}_i, \mathscr{E}_i, \mathscr{O}_i \rangle \Rightarrow \langle \mathscr{S}_{i+1}, \mathscr{E}_{i+1}, \mathscr{O}_{i+1} \rangle,$$

in the step from i to $i + 1$ $(0 \le i < m)$, the equations $l_1 \doteq r_1$ and $l_2 \doteq r_2$ are chosen in the set $\mathscr{E}_i^j$ such that $j \ge 1$ is the least index such that $F \doteq T \in \mathscr{E}_i^l$ for every $l < j$ and $F \doteq T \notin \mathscr{E}_i^j$.

In some sense, such derivations compute rigid $\vec{E}$-unifiers incrementally from left to right.

THEOREM 12.11 (INCREMENTAL COMPLETENESS). *Theorem 12.9 holds with lr-derivations instead of arbitrary derivations.*

This sharpening of Theorem 12.9 is very useful in practice, because it yields an incremental way of finding rigid $\vec{E}$-unifiers. From Theorem 12.9, it is obvious that Theorem 10.5 also holds for a family of sets of equations $\vec{E}$ and a term system $\mathscr{S}$.

THEOREM 12.12. *Let $\vec{E} = \{E_i \mid 1 \le i \le n\}$ and $S = \{\langle u_i, v_i \rangle \mid 1 \le i \le n\}$ as in Definition 12.1, $E^i = E_i \cup \{eq_i(u_i, v_i) \doteq F, eq_i(z_i, z_i) \doteq T\}$ for every i, $1 \le i \le n$, m the number of variables in $\vec{E} \cup S$, and $V = Var(\vec{E}) \cup Var(S)$. There is a finite complete set of rigid $\vec{E}$-unifiers for S given by the set*

$$\{\theta_{\mathscr{S}_k}|_V \mid \langle \mathscr{S}_0, \mathscr{E}_0, \mathscr{O}_0 \rangle \Rightarrow^+ \langle \mathscr{S}_k, \mathscr{E}_k, \mathscr{O}_k \rangle, \quad k \le m\},$$

for any order assignment $\mathscr{O}_0$ on $\langle E^1, \ldots, E^n \rangle$, with $\mathscr{S}_0 = \emptyset$, $\mathscr{E}_0^i = R(E^i, \mathscr{O}_0)$ for every i, $1 \le i \le n$, $\mathscr{E}_0 = \langle \mathscr{E}_0^1, \ldots, \mathscr{E}_0^n \rangle$, and where $\mathscr{S}_k$ is unifiable, $F \doteq T \in \mathscr{E}_k^j$, $F \doteq T \notin \mathscr{E}_i^j$ for all i and j, $0 \le i < k$, $1 \le j \le n$, and $\theta_{\mathscr{S}_k}$ is the mgu of $\mathscr{S}_k$ over $T_\Sigma(X)$.

Finally, it is obvious that Theorem 12.9 yields a generalization of Theorem 11.3 to equational prematings.

THEOREM 12.13. *Finding whether a pair* $\langle \vec{E}, S \rangle$ *(as in Definition* 12.1*) is an equational premating is NP-complete.*

As a consequence, since the problem of deciding whether a family of mated sets forms an equational mating is equivalent to the problem of finding whether a pair $\langle \vec{E}, S \rangle$ is an equational premating, the former problem is also NP-complete.

In the next section, we present a procedure based on the method of equational matings. The basic idea of such a procedure is straightforward, as suggested by Theorem 5.5: Compute incrementally amplifications of a formula in nnf, and at each stage, test whether such an amplification has a p-acceptable mating. The efficiency related issues here are the same as in Andrews's nonequational case, except that they are harder: In addition to efficient data structures that save information between stages, we must identify mated sets instead of mated pairs, and use rigid $\vec{E}$-unification instead of standard unification.

Although implementation issues are of importance for a practical procedure, we do not feel they are as new as the ideas of equational matings and rigid $\vec{E}$-unification, and thus, we only give a high-level description of the procedure.

13. *A Refutation Procedure*

We now consider defining a refutation procedure based on equational matings and rigid $\vec{E}$-unification. As mentioned in Section 5, such a procedure is suggested by Theorem 5.5: compute incrementally amplifications of a formula in nnf, and at each stage, test whether such an amplification has a p-acceptable mating. This idea can be formalized in the following nondeterministic definition, which uses the incremental rigid E-unification algorithm *E_UNIF* given by Theorem 12.11.

Definition 13.1. Let A be a rectified universal sentence in nnf and D an amplification of A. An *EQ-derivation* R is a sequence of tuples $\langle (D_0, \Pi_0, MS_0, \theta_0), \ldots, (D_p, \Pi_p, MS_p, \theta_p) \rangle$, such that for $0 \leq i \leq p$, D_i is an amplification of A, Π_i is a set of vertical paths in D_i, MS_i is a set of mated sets, each such set of the form $\{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n), \neg(s \doteq t)\}$, and θ_i is a substitution, such that

(1) D_0 is the quantifier-free form of A, $\Pi_0 = vp(D_0)$, $MS_0 = \varnothing$, and $\theta_0 = Id$, and

(2) For every i, $0 \leq i < p$, if $MS_i = \{S_1, \ldots, S_m\}$, then either

   (i) There is some vertical path π_{i+1} in Π_i, some subset S_{i+1} of π_{i+1} such that $S_{i+1} = \{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n), \neg(s \doteq t)\}$, and some rigid E-unifier σ_{i+1} for S_{i+1} given by the procedure *E_UNIF* (where $E = \{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n)\}$). Then $D_{i+1} = D_i$, $\Pi_{i+1} = \sigma_{i+1}(\Pi_i - \{\pi_{i+1}\})$, $MS_{i+1} = MS_i \cup \{S_{i+1}\}$, and $\theta_{i+1} = \theta_i; \sigma_{i+1}$; or

   (ii) If D_i is obtained from the rectified form of a sentence C_i by deleting quantifiers, where C_i is a sentence in a sequence $\langle C_1, \ldots, C_i \rangle$ $(i \geq 1)$ of formulas resulting from quantifier duplications, then D_{i+1} is obtained from the rectified form of a sentence C_{i+1}, obtained by quantifier

duplication from C_i, by deleting quantifiers. Then, $\Pi_{i+1} = vp(D_{i+1})$, $MS_{i+1} = \emptyset$, and $\theta_{i+1} = Id$.

If in addition, $\langle MS_p, \theta_p \rangle$ is a p-acceptable mating, we say that R is an *EQ-refutation.*

Since the method outlined in Definition 13.1 nondeterministically enumerates all equational matings for potentially all amplifications of A, it is immediate from Theorem 5.5, that since A is unsatisfiable iff some amplification of A has a p-acceptable mating, A is unsatisfiable iff there is an EQ-refutation for A.

There are a number of implementation problems with Definition 13.1 as the basis for a practical method for showing the unsatisfiability of a formula in nnf:

(1) The p-acceptable mating found is maximal, since no attempt is made to identify overlapping vertical paths that are spanned by common mated sets.
(2) Every time a subformula of A is amplified, the entire computation up until that point is discarded.
(3) No effort is made to use any failure in any step as a source of information for the next step.

These points are closely related to the acceptability criteria given by Andrews in Section 2.3 of [1], where he defines a procedure for finding (nonequational) p-acceptable matings. We tried to adapt the notion of a connection graph used by Andrews in Section 3 of [1], but unfortunately with no success. The difficulty in the presence of equality is that a vertical path π is closed iff it contains some mated *set* $\{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n), \neg(s \doteq t)\}$ such that s and t have some rigid E-unifier (where $E = \{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n)\}$), but there is no guarantee that $n \leq 1$. For languages without equality, a mated set is of the form $\{L, \neg L'\}$ where L and L' are unifiable. In order to determine which pairs of literals are unifiable it is necessary to examine $O(n^2)$ pairs, where n is the total number of literals in D. Hence, for languages without equality, it is advantageous to precompute a connection graph recording the pairs of literals $\{L, \neg L'\}$ where L and L' are unifiable, since every closed path must contain such a pair. However, for languages with equality, if D contains $n = q + r$ literals where q literals are positive and r literals are negative ($r \geq 1$), to form a mated set there are r choices for the negative literal and for each such choice, any subset of the positive literals can be chosen. Thus, there are potentially $r2^q$ mated sets, that is, an exponential number of mated sets. In addition, rigid E-unification is NP-complete.

Thus, the cost of determining which sets of literals are mated sets is exponential and there does not seem to be any advantage in computing such sets. Since our investigations on this subject are still very preliminary, we shall not elaborate any further. However, this is a very interesting topic that clearly requires more work.

We conclude this section with a naive procedure written in pseudocode implementing the method of equational matings. We have made no efforts towards improving efficiency of the basic method. This aspect should be addressed in further work.

Let us now turn our attention to identifying mated sets in the set $vp(D)$ of vertical paths in D. Since mated sets are of the form $\{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n), \neg(s \doteq t)\}$, the search is organized around the negative literals. Observe that if some path $\pi \in vp(D)$ does not contain a negated literal, it cannot

contain an unsatisfiable mated set and D is satisfiable. In this case, the procedure stops with failure. Suppose now that for some vertical path π, there is a mated set S of the form $\{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n), \neg(s \doteq t)\}$ but it has no rigid E-unifier. Then, we must perform an amplification step. We would like this step to supply information that was missing in the attempt to find an unsatisfiable mated set. Unfortunately, any arbitrary duplication may fail to do this, and may even introduce *new* vertical paths. As Andrews says in discussing this problem in the context of looking for nonequational p-acceptable matings, "One would like to develop a set of heuristics for duplicating quantifiers." However, it is beyond the scope of this paper to consider this issue in detail (as it was beyond the scope of Andrews and Bibel's papers), and we use a straightforward breadth-first outermost duplication strategy: In the lexical order of occurrences of subformulas in A, perform an outermost duplication of the first nonground subformula, then the second, and so on until the last nonground subformula in the lexical order has been duplicated, and then start again from the top. A breadth-first strategy clearly generates a complete search space of outermost duplications; however, it can clearly result in superfluous paths.[18]

Example 13.2. Consider the following formula A, formed from the union of Examples 4.8 and 5.6, where $\hat{x}, \hat{y}, \hat{z}, u, v, w, x, y, z$ denote variables:

$$(a \doteq b) \vee \neg(*(a,b) \doteq *(b,a)) \tag{10}$$

$$\wedge\ \forall\hat{x}\left(\left(f^3\hat{x} \doteq \hat{x}\right) \vee \neg\left(f\hat{x} \doteq fb\right)\right) \tag{11}$$

$$\wedge\ \left(Qa \vee \neg\left(f^3a \doteq a\right)\right) \tag{12}$$

$$\wedge\ \forall\hat{y}\left(\left(f^5\hat{y} \doteq \hat{y}\right) \vee \neg Q\hat{y}\right) \tag{13}$$

$$\wedge\ (Ra \vee \neg(fa \doteq a) \vee \neg Pfa) \tag{14}$$

$$\wedge\ \forall\hat{z} \neg Rf\hat{z} \tag{15}$$

$$\wedge\ Pa \tag{16}$$

$$\wedge\ \forall x\, \forall y\, \forall z(*(x, *(y,z)) \doteq *(*(x,y),z))) \tag{17}$$

$$\wedge\ \forall u(*(u,1) \doteq u) \tag{18}$$

$$\wedge\ \forall v(*(1,v) \doteq v) \tag{19}$$

$$\wedge\ \forall w(*(w,w) \doteq 1). \tag{20}$$

The problem is to find the right sequence of duplications for A. We know from Example 5.6 that 3 duplications of subformula (17) and 2 duplications of subformula (20) are necessary for the existence of a p-acceptable mating. But an obvious breadth-first outermost sequence of duplications results in 21 duplications before these duplications are generated. Subformulas (10), (12), (14), and (16) are ground and thus not subject to duplication.

Let A be a rectified universal formula in nnf. A breadth-first outermost amplification sequence $\langle D_1, \ldots, D_i \rangle$ $(i \geq 1)$ for A is a sequence such that D_k is obtained from the rectified form of a sentence C_k by deleting quantifiers, where C_k is a sentence in a sequence $\langle C_1, \ldots, C_i \rangle$ $(i \geq 1)$ of formulas resulting

[18] Andrews and Bibel have shown that outermost duplication itself can generate superfluous literals.

from quantifier duplications ($1 \leq k \leq i$). The C_k's are defined such that $C_1 = A$, and C_{k+1} is obtained by quantifier duplication of an outermost universal subformula of C_k using a breadth-first strategy ($1 \leq k \leq i-1$). Given a counter $k \geq 1$, a call *amplify*$(k+1, A, D)$ to the procedure *amplify* returns the amplification $D = D_{k+1}$ obtained from C_k as explained above. For $k = 0$, it is assumed that *amplify*$(1, A, D)$ returns the quantifier-free formula D_1 obtained by deleting quantifiers from A. We are assuming that the $\vec{E}$-unification algorithm *E_UNIF* (given by Theorem 12.11) takes as input a mated set $S = \{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n), \neg(s \doteq t)\}$ and returns a finite complete set of rigid E-unifiers of s and t where $E = \{(s_1 \doteq t_1), \ldots, (s_n \doteq t_n)\}$, in the form of a set of triangular forms, where each triangular form T represents a substitution σ_T. For presenting the refutation procedure, we also assume that a mating $\mathcal{M}$ is represented by a pair $\langle MS, U \rangle$, where MS is a collection of mated sets, and U is the triangular representation of a substitution. Also, given a set of paths Π and a substitution σ, let

$$apply(\sigma, \Pi) = \{\pi' \mid \pi = \{P_1, \ldots, P_k\} \in \Pi \text{ and } \pi' = \{\sigma(P_1), \ldots, \sigma(P_k)\}\}.$$

We assume that the application of σ to Π is done intelligently, that is, since σ is the identity substitution on almost all literals in π, some table lookup mechanism is available to identify the literals that have variables in the domain of σ.

We collect the information discussed in this section into the following pseudo-code version of a refutation procedure for formulas in nnf that uses the following variables and procedures: A is a rectified universal sentence in nnf; i is a counter for vertical paths in an amplification; j is a counter for negative literals in a vertical path; k is an amplification counter; $\mathcal{M}$ is an equational mating; *p_acceptable* is a Boolean value which is true iff $\mathcal{M}$ is p_acceptable; *found* is a Boolean value that is true if an unsatisfiable mated set is identified in some path; *select_path*$(i, vp(D))$ returns the ith path associated with amplification D; *select_negative_literal*(j, π) returns the jth negative literal in π; *choose_positive_subset*(π) returns the set of positive literals for some path π. This procedure must be understood as a nondeterministic procedure. A deterministic version can be written by implementing explicitly the backtracking needed to handle the choice of literal, path, etc. However, we feel that it would not be as clear as the present version.

A Refutation Procedure

```
procedure equational_refutation(A);
begin
  k ← 1;
  amplify(k, A, D);  (D is thus the quantifier-free form of A)
  M ← ⟨∅, ∅⟩;
  i_limit ← #paths(vp(D)); i ← 1;
  p_acceptable ← ⊥ ;
  while i ≤ i_limit ∧ ¬ p_acceptable do
    π ← select_path(i, vp(D));
    j_limit ← #negative_literals(π); found ← ⊥ ;
    if j_limit ≠ 0 then
      [j ← 1;
        while j ≤ j_limit ∧ ¬ found do
```

```
            N ← select_negative_literal(j, π);
            S ← choose_positive_subset(π) ∪ {N};
            if ∃T ∈ E_UNIF(S) then
              [found ← ⊤ ;
              if i = i_limit then p_acceptable ← ⊤ else i ← i + 1;
              let ⟨MS, U⟩ = 𝓜; 𝓜 ← ⟨MS ∪ S, U ∪ T⟩;
              vp(D) ← apply(σ_T, vp(D))]
            else j ← j + 1
          endwhile]
      else return; {A is satisfiable}
      if ¬ found then
        [k ← k + 1;
         amplify(k, A, D);
         i_limit ← #paths(vp(D)); i ← 1]
    endwhile;
    return(𝓜)
end;
```

14. *Conclusion and Further Work*

We have generalized Andrews and Bibel's method of matings to first-order languages with equality. This new method is sound and complete, and uses a decidable form of $\vec{E}$-unification, rigid $\vec{E}$-unification. We have shown that both rigid $\vec{E}$-unification and finding whether a pair $\langle \vec{E}, S \rangle$ is an equational premating are NP-complete problems. We also have shown that finite complete sets of rigid $\vec{E}$-unifiers always exist. Theorem 12.13 has important implications regarding the computational complexity of theorem proving for first-order languages with equality using the method of matings. It shows that there is an algorithm for finding equational matings, but not only is the problem of deciding whether an equational mating is p-acceptable co-NP-complete, the problem of deciding that a family of mated sets is an equational mating is NP-complete. For languages without equality, the first problem is still co-NP-complete, but the second can be solved in polynomial time using standard unification, and in fact in linear time.

It is essential to find ways of trimming the search space of order assignments if we want the method to be practical. When a reduction ordering $\prec$ is available and all subterms in $\mathscr{E}_t'$ are ordered by $\prec$, $\mathscr{O}_t$ is completely determined. It would be interesting to investigate subcases where order assignments can be found quickly. An actual implementation of the refutation procedure would also be interesting, as well as a comparison with other methods, those based on Knuth–Bendix completions in particular. The above questions are left for further research.

Appendix. Proof of the Skolem–Herbrand–Gödel Theorem

In this section, we give a semantic proof of the Skolem–Herbrand–Gödel theorem, in the line of Andrews's proof [1, 2]. The proof relies on two properties:

(1) If every c-instance of a universal sentence A in nnf is satisfiable, then the set of all c-instances of A is satisfiable. This follows from an easy application of the compactness theorem, as in Andrews [1, 2].

(2) If a universal sentence in nnf (with equality) is valid in some model **M**, then it is valid in some model $\mathscr{H}$ whose domain is the quotient of the Herbrand universe by some congruence.

For languages without equality, property (2) is simpler. If a sentence is valid in some model, then it is valid in some Herbrand model, and there is no need for a quotient construction. We now proceed with the proofs.

LEMMA A1. *Let A be a universal sentence in nnf. If every c-instance of A is satisfiable, then the set of all c-instances of A is satisfiable.*

PROOF. First, we use the fact proved in Andrews [1, Lemma 1] or Gallier [17, Lemma 7.6.1], that for any two c-instances K and L of A, there is some c-instance D of A such that $\models D \supset (K \wedge L)$. Then, assume that every c-instance of A is satisfiable. For every finite set $\{K_1, \ldots, K_n\}$ of c-instances of A, using the above property $n - 1$ times, we have some c-instance K of A, such that, $\models K \supset (K_1 \wedge \cdots \wedge K_n)$. Since every c-instance of A is satisfiable, the set $\{K_1, \ldots, K_n\}$ is satisfiable. By the compactness theorem, the set of all c-instances of A is satisfiable. □

LEMMA A2. *Consider a first-order language with equality having at least one constant. Given a sentence A in negation normal form and not containing existential quantifiers, if A is satisfied in some structure, then A is satisfied in some structure whose domain is the quotient of the Herbrand universe HT by some congruence $\simeq$.*

PROOF (SKETCH). Assume that $\mathbf{M} \models A$, for some structure $\mathbf{M}$. Let $\mathscr{HT}$ be the initial algebra[19] generated by the constant and function symbols in the language (whose domain is the Herbrand universe HT). Let h be the unique algebra homomorphism $h: \mathscr{HT} \to \mathbf{M}$ defined such that:

For every constant c, $h(c) = c_{\mathbf{M}}$;
For every function symbol f of rank $n > 0$, for any n terms $t_1, \ldots, t_n \in HT$,

$$h(ft_1, \ldots, t_n) = f_{\mathbf{M}}(h(t_1), \ldots, h(t_n)).$$

It is immediate by the definition of h that for every term $t \in HT$, $t_{\mathbf{M}} = h(t)$.
Let $\simeq$ be the kernel of the homomorphism h, that is, the relation on HT defined such that, for all $s, t \in HT$, $s \simeq t$ iff $h(s) = h(t)$. It is well known that $\simeq$ is a congruence on $\mathscr{HT}$. Observe that $s \simeq t$ iff $\mathbf{M} \models (s \doteq t)$, since $s_{\mathbf{M}} = h(s)$ and $t_{\mathbf{M}} = h(t)$. Let $\mathscr{H}$ be the quotient algebra $\mathscr{HT}/\simeq$. Since $\simeq = \mathit{kernel}(h)$, there is a unique homomorphism $\bar{h}: \mathscr{H} \to \mathbf{M}$, such that $\overline{h(t)} = h(t)$, for every $\bar{t} \in HT/\simeq$.
We make $\mathscr{H}$ into a structure as follows: For each predicate symbol P of rank n, for any n equivalence classes of term $\bar{t}_1, \ldots, \bar{t}_n \in HT/\simeq$,

$$P_{\mathscr{H}}(\bar{t}_1, \ldots, \bar{t}_n) = \mathbf{true} \quad \text{iff} \quad P_{\mathbf{M}}(h(t_1), \ldots, h(t_n)) = \mathbf{true}.$$

Note that for every $t \in HT$, we have

$$t_{\mathbf{M}} = \bar{h}(\bar{t}),$$

since $\bar{h}(\bar{t}) = h(t)$ and $t_{\mathbf{M}} = h(t)$.
Given a formula A with set of free variables $\{x_1, \ldots, x_n\}$, and a structure $\mathbf{M}$, for any n-tuple $(m_1, \ldots, m_n) \in M^n$, $\mathbf{M} \models A[m_1, \ldots, m_n]$ means that $\mathbf{M} \models A[s]$ for any assignment s such that $s(x_i) = m_i$, for $1 \leq i \leq n$. (It is well

[19] For details on algebras and homomorphisms, see Gallier [17].

known Gallier [17] that $A_{\mathbf{M}}[s]$ only depends on the restriction of s to $\{x_1, \ldots, x_n\}$). The following properties can be proved by induction on terms and formulas:

(1) For every term t with free variables $\{x_1, \ldots, x_n\}$, for every n-tuple, $\overline{t_1}, \ldots, \overline{t_n} \in HT/\simeq$,

$$t_{\mathscr{H}}(\overline{t_1}, \ldots, \overline{t_n}) = \overline{t[t_1/x_1, \ldots, t_n/x_n]};$$

(2) For every atomic formula B (including the case of an equation) with free variables $\{x_1, \ldots, x_n\}$, for every n-tuple, $\overline{t_1}, \ldots, \overline{t_n} \in HT/\simeq$

$$\mathscr{H} \models B[\overline{t_1}, \ldots, \overline{t_n}] \quad \text{iff} \quad \mathscr{H} \models B[t_1/x_1, \ldots, t_n/x_n].$$

Using induction on formulas, we shall establish the following claim:

Claim. For every formula X in negation normal form and not containing any existential quantifiers, for every assignment a: $\mathscr{V} \to HT/\simeq$, if $\mathbf{M} \models X[a \circ \bar{h}]$, then $\mathscr{H} \models X[a]$. The proof is similar to that in Gallier [17].

Proof of claim. We proceed by induction on formulas.

(i) First, assume that X is an equation $(s \doteq t)$, with set of free variables $\{x_1, \ldots, x_n\}$, and that for some n-tuple $(t_1, \ldots, t_n) \in HT^n$, we have

$$\mathbf{M} \models (s \doteq t)[\bar{h}(\overline{t_1}), \ldots, \bar{h}(\overline{t_n})].$$

Since for every $t \in HT$, $t_{\mathbf{M}} = \overline{h(t)}$, we have

$$\begin{aligned} s_{\mathbf{M}}(\bar{h}(\overline{t_1}), \ldots, \bar{h}(\overline{t_n})) &= s_{\mathbf{M}}((t_1)_{\mathbf{M}}, \ldots, (t_n)_{\mathbf{M}}) \\ &= (s[t_1/x_1, \ldots, t_n/x_n])_{\mathbf{M}} \quad \text{and} \\ t_{\mathbf{M}}(\bar{h}(\overline{t_1}), \ldots, \bar{h}(\overline{t_n})) &= t_{\mathbf{M}}((t_1)_{\mathbf{M}}, \ldots, (t_n)_{\mathbf{M}}) \\ &= (t[t_1/x_1, \ldots, t_n/x_n])_{\mathbf{M}}. \end{aligned}$$

Hence, the hypothesis $\mathbf{M} \models (s \doteq t)[\bar{h}(\overline{t_1}), \ldots, \bar{h}(\overline{t_n})]$ is equivalent to

$$\mathbf{M} \models (s[t_1/x_1, \ldots, t_n/x_n] \doteq t[t_1/x_1, \ldots, t_n/x_n]).$$

By the definition of $\simeq$, this shows that

$$s[\overline{t_1/x_1, \ldots, t_n/x_n}] = t[\overline{t_1/x_1, \ldots, t_n/x_n}]. \qquad (*)$$

Since for every $\overline{t_1}, \ldots, \overline{t_n} \in HT/\simeq$, we have

$$\begin{aligned} s_{\mathscr{H}}(\overline{t_1}, \ldots, \overline{t_n}) &= s[\overline{t_1/x_1, \ldots, t_n/x_n}] \quad \text{and} \\ t_{\mathscr{H}}(\overline{t_1}, \ldots, \overline{t_n}) &= t[\overline{t_1/x_1, \ldots, t_n/x_n}], \end{aligned}$$

by $(*)$, we have shown that

$$\mathbf{M} \models (s \doteq t)[\bar{h}(\overline{t_1}), \ldots, \bar{h}(\overline{t_n})] \quad \text{iff} \quad \mathscr{H} \models (s \doteq t)[\overline{t_1}, \ldots, \overline{t_n}].$$

(ii) If $X = Ps_1 \dots s_m$, with set of free variables $\{x_1, \dots, x_n\}$, we have

$$\begin{aligned} & X_{\mathbf{M}}\big(\overline{h}(\overline{t_1}), \dots, \overline{h}(\overline{t_n})\big) \\ &\quad = X_{\mathbf{M}}((t_1)_{\mathbf{M}}, \dots, (t_n)_{\mathbf{M}}) \\ &\quad = (X[t_1/x_1, \dots, t_n/x_n])_{\mathbf{M}} \\ &\quad = (Ps_1[t_1/x_1, \dots, t_n/x_n], \dots, s_m[t_1/x_1, \dots, t_n/x_n])_{\mathbf{M}} \\ &\quad = P_{\mathbf{M}}((s_1[t_1/x_1, \dots, t_n/x_n])_{\mathbf{M}}, \dots, (s_m[t_1/x_1, \dots, t_n/x_n])_{\mathbf{M}}) \\ &\quad = P_{\mathbf{M}}(h(s_1[t_1/x_1, \dots, t_n/x_n]), \dots, h(s_m[t_1/x_1, \dots, t_n/x_n])), \end{aligned}$$

and

$$\begin{aligned} X_{\mathscr{H}}(\overline{t_1}, \dots, \overline{t_n}) &= (X[t_1/x_1, \dots, t_n/x_n])_{\mathscr{H}} \\ &= (Ps_1[t_1/x_1, \dots, t_n/x_n], \dots, s_m[t_1/x_1, \dots, t_n/x_n])_{\mathscr{H}} \\ &= P_{\mathscr{H}}((s_1[t_1/x_1, \dots, t_n/x_n])_{\mathscr{H}}, \dots, (s_m[t_1/x_1, \dots, t_n/x_n])_{\mathscr{H}}) \\ &= P_{\mathscr{H}}\big(s_1\big[\overline{t_1/x_1, \dots, t_n/x_n}\big], \dots, s_m\big[\overline{t_1/x_1, \dots, t_n/x_n}\big]\big). \end{aligned}$$

Since for any n terms $\overline{r_1}, \dots, \overline{r_n} \in HT/\simeq$,

$$P_{\mathscr{H}}(\overline{r_1}, \dots, \overline{r_n}) = \mathbf{true} \quad \text{iff} \quad P_{\mathbf{M}}(h(r_1), \dots, h(r_n)) = \mathbf{true},$$

then

$$\mathscr{H} \models X\big[\overline{t_1}, \dots, \overline{t_n}\big] \quad \text{iff} \quad \mathbf{M} \models X\big[\overline{h}(\overline{t_1}), \dots, \overline{h}(\overline{t_n})\big].$$

(iii) If $X = \neg B$, where B is an atomic formula, the result holds because we have shown equivalences in (i) and (ii).

(iv) If X is of the form $(B \wedge C)$, then $\mathbf{M} \models X[a \circ \overline{h}]$ implies that

$$\mathbf{M} \models B\big[a \circ \overline{h}\big] \quad \text{and} \quad \mathbf{M} \models C\big[a \circ \overline{h}\big].$$

By the induction hypothesis,

$$\mathscr{H} \models B[a] \quad \text{and} \quad \mathscr{H} \models C[a],$$

that is, $\mathscr{H} \models X[a]$.

(v) If X is of the form $(B \vee C)$, then the proof is similar to case (iv).

(vi) X is of the form $\exists x B$. This case is not possible since X does not contain existential quantifiers.

(vii) X is of the form $\forall x B$. If $\mathbf{M} \models X[a \circ \overline{h}]$, then for every $m \in M$,

$$\mathbf{M} \models B\big[\big(a \circ \overline{h}\big)[x := m]\big].$$

(Given an assignment a, the notation $a[x := m]$ denotes the assignment a' such that $a'(x) = m$, and $a'(y) = a(y)$ for all $y \neq x$). Now, since $\overline{h}: \mathscr{H} \to \mathbf{M}$, for every $\bar{t} \in H$, $\overline{h}(\bar{t}) \in M$, and so, for every $\bar{t} \in H$,

$$\mathbf{M} \models B\big[\big(a \circ \overline{h}\big)\big[x := \overline{h}(\bar{t})\big]\big], \quad \text{that is,} \quad \mathbf{M} \models B\big[\big(a[x := \bar{t}]\big) \circ \overline{h}\big].$$

By the induction hypothesis, $\mathscr{H} \models B[a[x := \bar{t}]]$ for all $\bar{t} \in H$, that is, $\mathscr{H} \models X[a]$. This concludes the proof of the claim. □

From the claim, since **M** is a model of A, we have shown that $\mathscr{H}$ is a model of A. □

It is clear that Lemma A2 also holds for sets of universal sentences in nnf. Finally, we prove the Skolem–Herbrand–Gödel theorem.

THEOREM 5.2. *Given a universal sentence A in nnf, A is unsatisfiable iff some c-instance C of A is unsatisfiable.*

PROOF. First, assume that some compound instance C is unsatisfiable. It is straightforward to show that $\models A \supset C$ (see Gallier [17, Theorem 7.6.1]). Hence, A is unsatisfiable.

To establish the converse, we prove its contrapositive: If every c-instance of A is satisfiable, then A is satisfiable.

Since every c-instance of A is satisfiable, by Lemma A1, the set of all c-instance of A is satisfiable. By Lemma A2 (extended to sets of sentences), the set of all c-instances is valid in some structure $\mathscr{H}$ whose domain is the quotient of the Herbrand universe HT by some congruence $\simeq$. We prove by induction on the structure of A that A is valid in $\mathscr{H}$.

Case 1. A is a literal. Then, A is the only c-instance of A, and the result holds since $\mathscr{H}$ is a model of all c-instances of A.

Case 2. $A = (B \wedge C)$. Let K be a c-instance of B, and L be a c-instance of C. Then, $(K \wedge L)$ is a c-instance of A. Since $\mathscr{H}$ is a model of all c-instances of A, we have $\mathscr{H} \models (K \wedge L)$, that is, since K and L are ground formulas, $\mathscr{H} \models K$ and $\mathscr{H} \models L$. By the induction hypothesis, $\mathscr{H} \models B$ and $\mathscr{H} \models C$, which, since A is a sentence, implies that $\mathscr{H} \models A$.

Case 3. $A = (B \vee C)$. We claim that either $\mathscr{H}$ is a model of all c-instances of B, or that $\mathscr{H}$ is a model of all c-instances of C. Indeed, if this was not the case, there would be some c-instance K of B and some c-instance L of C such that $\mathscr{H} \not\models B$ and $\mathscr{H} \not\models C$. However, since $(K \vee L)$ is a c-instance of A, we would have $\mathscr{H} \not\models (K \vee L)$, contradicting the fact that $\mathscr{H}$ is a model of all c-instances of A. Thus, by the induction hypothesis, either $\mathscr{H} \models B$ or $\mathscr{H} \models C$, which, since A is a sentence, implies that $\mathscr{H} \models A$.

Case 4. $A = \forall x B$. Let t be any (ground) term in HT. Every c-instance of $B[t/x]$ is a c-instance of A, and since $\mathscr{H}$ is a model of every c-instance of A, by the induction hypothesis, we have $\mathscr{H} \models B[t/x]$. However, in the proof of Lemma A2, we have shown:

Fact. For every atomic formula B (including the case of an equation) with free variables $\{x_1, \ldots, x_n\}$, for every n-tuple, $\overline{t_1}, \ldots, \overline{t_n} \in HT/\simeq$,

$$\mathscr{H} \models B\left[\overline{t_1}, \ldots, \overline{t_n}\right] \quad \text{iff} \quad \mathscr{H} \models B[t_1/x_1, \ldots, t_n/x_n].$$

By a straightforward induction on formulas almost identical to the proof of the claim in Lemma A2, we can generalize the above fact to any universal formula B in nnf. But then, $\mathscr{H} \models B[t/x]$ iff $\mathscr{H} \models B[\bar{t}]$, where $[\bar{t}]$ denotes any assignment $s[x := \bar{t}]$ such that $s(x) = \bar{t}$. Hence, for every $\bar{t} \in HT$, we have $\mathscr{H} \models B[\bar{t}]$, and by the semantics of quantifiers, this means that $\mathscr{H} \models \forall x B$. Therefore, $\mathscr{H} \models A$, as desired. □

ACKNOWLEDGMENT. We wish to thank Peter Andrews, Leo Bachmair, Jin Choi, Jean Yves Girard, Tomas Isakowitz, Dale Miller, Frank Pfenning, David Plaisted, and Richard Statman for some helpful suggestions and for their encouragement.

REFERENCES

1. ANDREWS, P. B. Theorem proving via general matings. *J. ACM 28*, 2 (Apr. 1981), 193–214.
2. ANDREWS, P. B. *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof*. Academic Press, Orlando, Fla., 1986.
3. ANDREWS, P. B., MILLER, D., COHEN, E., AND PFENNING, F. Automating higher-order logic, *Contemp. Math. 29*, (1984), 169–192.
4. BACHMAIR, L. Canonical equational proofs. In *Research Notes in Theoretical Computer Science*. Wiley, New York, 1989.
5. BACHMAIR, L., DERSHOWITZ, N., AND PLAISTED, D. Completion without failure. In *Resolution of Equations in Algebraic Structures*, vol. 2. In Aït-Kaci and Nivat, eds. Academic Press, 1989, pp. 1–30.
6. BACHMAIR, L., DERSHOWITZ, N., AND HSIANG, J. Orderings for equational proofs. In *Logic in Computer Science (LICS'86)* (Cambridge, Mass., June 16–18). IEEE, New York, 1986, pp. 346–357.
7. BIBEL, W. Tautology testing with a generalized matrix reduction method. *Theoret. Comput. Sci. 8* (1979), 31–44.
8. BIBEL, W. On matrices with connections. *J. ACM 28*, 4 (Oct. 1981), 633–645.
9. BIBEL, W. *Automated Theorem Proving*. Friedrich Vieweg & Sohn, Braunschweig, Germany, 1982.
10. BIBEL, W., AND SCHREIBER, J. Proof search in a Gentzen-like system of first-order logic. In *Proceedings of the International Computing Symposium*. North-Holland, Amsterdam, 1975, pp. 205–212.
11. DERSHOWITZ, N. Termination of rewriting. *J. Symb. Comput. 3* (1987), 69–116.
12. DERSHOWITZ, N., MARCUS, L., AND TARLECKI, A. Existence, uniqueness, and construction of rewrite systems. *SIAM J. Comput. 17* (1988), 629–639.
13. DOWNEY, P. J., SETHI, R., AND TARJAN, R. E. Variations on the common subexpression problem. *J. ACM 27*, 4 (Oct. 1980), 758–771.
14. EDER, E. A comparison of the resolution and the connection method. In E. Börger, H. Kleine Büning, and M. M. Richter, eds., *Computer Science and Logic* (CSL'88), *2nd Workshop on Computer Science Logic*. Springer-Verlag, New York, 1989, pp. 80–98.
15. EDER, E. Relative complexities of first-order calculi. Habilitationsschrift. Universität Dortmund, Germany, 1990. To appear as a book in Vieweg Verlag Braunschweig.
16. FITTING, M. A tableaux based automated theorem prover for classical logic. Tech. Rep. City Univ. of New York, New York, June 1986.
17. GALLIER, J. H. *Logic for Computer Science: Foundations of Automatic Theorem Proving*. Harper and Row, New York, 1986.
18. GALLIER, J. H., RAATZ, S., AND SNYDER, W. Theorem proving using rigid E-unification: Equational matings. In *Logic in Computer Science (LICS'87)* (Ithaca, N.Y.). IEEE, New York, 1987, pp. 338–346.
19. GALLIER, J. H., NARENDRAN, P., PLAISTED, D., RAATZ, S., AND SNYDER, W. Finding canonical rewriting systems equivalent to a finite set of ground equations in polynomial time. *J. ACM*, to appear.
20. GALLIER, J. H., NARENDRAN, P., PLAISTED, D., AND SNYDER, W. Rigid E-unification is NP-complete. In *Logic in Computer Science (LICS'88)* (Edinburgh, Scotland, July 5–8). IEEE, New York, 1988, pp. 218–227.
21. GALLIER, J. H., AND SNYDER, W. Complete sets of transformations for general E-unification. *Theoret. Comput. Sci. 67*, 2–3 (1989), 203–260.
22. GALLIER, J. H., NARENDRAN, P., PLAISTED, D., AND SNYDER, W. Rigid E-unification: NP-completeness and applications to theorem proving. *Inf. Comput. 87*, 1/2 (1990), 129–195.
23. GIRARD, J. Y. Linear logic. *Theor. Comput. Sci. 50*, 1 (1987), 1–102.
24. HERBRAND, J. *Logical Writings*. Reidel Publishing Company, Hingham, Mass., 1971.
25. HUET, G. A unification algorithm for typed λ-calculus. *Theor. Comp. Sci. 1* (1975), 27–57.

26. HUET, G. *Résolution d'Equations dans les Langages d'Ordre* $1, 2, \ldots, \omega$. Thése d'Etat, Université de Paris VII, Paris, France, 1976.
27. HUET, G. Confluent reductions: Abstract properties and applications to term rewriting systems. *J. ACM 27*, 4 (Oct. 1983), 797–821.
28. HUET, G., AND OPPEN, D. C. Equations and rewrite rules: A survey. In R. V. Book, ed., *Formal Languages: Perspectives and Open Problems*. Academic Press, Orlando, Fla., 1982, pp. 349–405.
29. KNUTH, D. E., AND BENDIX, P. B. Simple word problems in universal algebras. In J. Leech, ed., *Computational Problems in Abstract Algebra*, Pergamon Press, 1970, pp. 263–297.
30. KOZEN, D. Complexity of finitely presented algebras. Tech. Rep. TR 76-294. Dept. Computer Sci., Cornell Univ., Ithaca, N.Y., 1976.
31. KOZEN, D. Positive first-order logic is NP-complete. *IBM J. Res. Dev. 25*, 4 (1981), 327–332.
32. LEVY, A. *Basic Set Theory*. Springer-Verlag, New York, 1979.
33. MARTELLI, A., AND MONTANARI, U. An efficient unification algorithm. *ACM Trans. Prog. Lang. Syst. 4*, 2 (Apr. 1982), 258–282.
34. METIVIER, Y. About the rewriting systems produced by the Knuth–Bendix completion algorithm. *Inf. Proc. Lett. 16* (1983), 31–34.
35. MILLER, D. A. Expansion trees and their conversion to natural deduction proofs. In R. E. Shostak, ed., *Proceedings of the 7th International Conference on Automated Deduction* (Napa, CA). Lecture Notes in Computer Science, vol. 170. Springer-Verlag, New York, 1984, pp. 375–393.
36. NELSON, G., AND OPPEN, D. C. Fast decision procedures based on congruence closure. *J. ACM 27*, 2 (1980), 356–364.
37. PATERSON, M. S., AND WEGMAN, M. N. Linear unification. *J. Comput. Syst. Sci. 16* (1978), 158–167.
38. PFENNING, F. Proof transformations in higher-order logic. Ph.D. dissertation. Department of Mathematics, Carnegie Mellon Univ., Pittsburgh, Pa., Jan. 1987.
39. PLOTKIN, G. Building in equational theories. *Machine Intelligence 7* (1972), 73–90.
40. SMULLYAN, R. M. *First-order Logic*. Springer-Verlag, New York, 1968.
41. SZABO, M. E. The collected papers of Gerhard Gentzen. In: *Studies in Logic*. Elsevier North-Holland, New York, 1970.

RECEIVED JANUARY 1989; REVISED FEBRUARY AND JULY 1990; ACCEPTED JULY 1990