2 Completion and Its Applications

NACHUM DERSHOWITZ[1]

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois

1. Introduction

The ability to reason with equations is important in many computer science applications, including algebraic specifications, high-level programming languages, and automated deduction. Reasoning about equations may, for example, involve deciding if an equation "follows" as a consequence of a given set of equations, or if an equation is "true" in a given model, or what values of the variables satisfy a given equation. Knuth's completion procedure (Knuth and Bendix, 1970), based on prior work of Evans (1951), was originally suggested as a means of taking an axiomatization of an equational theory and generating a rewrite system that can be used to decide questions of validity of identities (or the "word problem") in the given theory. In this chapter, we formalize that procedure as an inference system and describe some of its more recent applications to other aspects of equational reasoning.

[1]This research was supported in part by the National Science Foundation under Grant DCR 85-13417.

ISBN 0-12-046371-7

1.1. Equations

Let $\mathcal{T}$ denote a set of (first-order) *terms* built out of function symbols taken from a vocabulary (signature) $\mathcal{F}$ and variables from a set $\mathcal{V}$. An *equational system* E over $\mathcal{T}$ is a set of identities, each of which we write here in the form $l \leftrightarrow r$, where l and r are terms in $\mathcal{T}$ and variables appearing in them are universally quantified. For convenience, we identify any equation $l \leftrightarrow r$ with $r \leftrightarrow l$; thus, E may be thought of as a symmetric binary relation over $\mathcal{T}$. Equations are applied to terms by replacing a subterm matching one side of an equation with its other side. More precisely, an equation $l \leftrightarrow r$ in E may be applied to a term t in $\mathcal{T}$, if there is some substitution σ of terms in $\mathcal{T}$ for variables in the equation, such that $l\sigma$ (the result of applying σ to l) is the same as some subterm s of t. Let t/p denote the subterm s of t rooted at position p within t. The equation is applied by replacing the subterm $t/p = l\sigma$ of t with the other side of the equation, $r\sigma$, after the same substitution σ of terms for variables has been made. The result of this replacement of "equals by equals" is denoted $t[r\sigma]_p$. We write $s \leftrightarrow_E t$ or $t \leftrightarrow_E s$ to indicate that the term t in $\mathcal{T}$ is obtainable from the term s in $\mathcal{T}$ by a single application of some equation in E. A *proof* in E is any finite sequence $t_1 \leftrightarrow_E t_2 \leftrightarrow_E \cdots \leftrightarrow_E t_n$ $(n \geq 1)$ of applications of equations in E. We write $E \dashv s = t$ if there is a proof $s \leftrightarrow^* t$ of $s = t$ in E. The *equational theory* (or *variety*) E is the class of equations provable in E. By a theorem of Birkhoff (1935), $s = t$ is provable in this way if, and only if, it is true in all models of E, i.e., if $s = t$ is valid for E. In symbols, $E \models s = t$ if and only if $E \dashv s = t$.

For example, the following is an equational system E for an operation with an identity and inverse:

$$\begin{array}{rcl} x \cdot 1 & \leftrightarrow & x \\ 1 \cdot x & \leftrightarrow & x \\ y^- \cdot (y \cdot z) & \leftrightarrow & z \end{array}$$

where x, y, and z are variables.[2] The identity

$$x \cdot x^- = 1$$

[2]Throughout this chapter, we will use these three letters for variables appearing in terms.

lends itself to the following six-step proof in E:

$$\begin{aligned} x \cdot x^{-} &\leftrightarrow_E (x^{--} \cdot (x^{-} \cdot x)) \cdot x^{-} \\ &\leftrightarrow_E (x^{--} \cdot (x^{-} \cdot (x \cdot 1))) \cdot x^{-} \\ &\leftrightarrow_E (x^{--} \cdot 1) \cdot x^{-} \\ &\leftrightarrow_E x^{--} \cdot x^{-} \\ &\leftrightarrow_E x^{--} \cdot (x^{-} \cdot 1) \qquad \leftrightarrow_E \quad 1 \end{aligned}$$

1.2. Rewrite Systems

A *rewrite* (or *term-rewriting) system* R over a set of terms $\mathcal{T}$ is a set of *directed* equations, called *rewrite rules*, each of the form $l \to r$, where l and r are terms in $\mathcal{T}$. Thus, R is a binary relation over $\mathcal{T}$. A rule $l \to r$ may be applied to a term t in $\mathcal{T}$ if a subterm s at some position p in t matches the left-hand side l via some substitution σ. Like equations, a rule is applied by replacing in t the subterm $s = l\sigma = t/p$ with the corresponding right-hand side $r\sigma$ of the rule, resulting in $t[r\sigma]_p$. We write $s \to_R t$ or $t \leftarrow_R s$ to indicate that the term s in $\mathcal{T}$ *rewrites* to the term t in $\mathcal{T}$ by a single application of some rule in R, the direction of the arrow distinguishing between applications of rules from left to right and from right to left, respectively. A *derivation* in R is any (finite or infinite) sequence $t_1 \to_R t_2 \to_R \cdots$ of directed applications of rules in R; a *proof* in R is any finite sequence $t_1 \leftrightarrow_R t_2 \leftrightarrow_R \cdots \leftrightarrow_R t_n$ $(n \geq 0)$, where each step $t_i \leftrightarrow_R t_{i+1}$ is a rewrite either from left to right or from right to left.

If a term t cannot be rewritten, we say that it is *irreducible*. If $s \to_R^* t$ and t is irreducible, then we write $s \to_R^! t$ and say that t is an *R-normal form* of s, or that s *reduces* to t. If every term in $\mathcal{T}$ reduces to an R-normal form, then R is said to be *normalizing*. In practice, one is usually interested in *terminating* systems, those for which infinite derivations $t_1 \to_R t_2 \to_R \ldots$ of terms $t_i \in \mathcal{T}$ are impossible. Only terminating rewrite systems are considered in this chapter; such systems are also normalizing. For finite, terminating R, the reduction relation $\to_R^!$ is decidable.

The following is an eight-rule system for multiplicative identity and inverse (a non-associative fragment of group theory):

$1 \cdot x$	$\rightarrow$	x	$x \cdot 1$	$\rightarrow$	x
$x^- \cdot x$	$\rightarrow$	1	$x \cdot x^-$	$\rightarrow$	1
1^-	$\rightarrow$	1	$(x^-)^-$	$\rightarrow$	x
$y^- \cdot (y \cdot z)$	$\rightarrow$	z	$y \cdot (y^- \cdot z)$	$\rightarrow$	z

Applying those rules to the term $(a^{--} \cdot (a^- \cdot (a \cdot 1))) \cdot a^-$ gives the graph of possible derivations shown in Fig. 1. In any case, the final result is 1 and no further applications of rules are possible, i.e., 1 is its *unique* normal form.

1.3. Deciding Validity

For any binary relation $\rightarrow$, we use the notations $\leftarrow$, $\leftrightarrow$, $\rightarrow^+$, and $\rightarrow^*$ to denote its inverse, symmetric closure, transitive closure, and reflexive-transitive closure, respectively. A binary relation $\rightarrow$ has the *Church-Rosser* property, if $\leftrightarrow^*$ is contained in $\rightarrow^* \circ \leftarrow^*$, where $\circ$ denotes composition of relations; it is *confluent* if $\leftarrow^* \circ \rightarrow^*$ is contained in $\rightarrow^* \circ \leftarrow^*$. These two properties are equivalent (Curry and Feys, 1958). We say that a binary relation $\rightarrow$ over a set of terms $\mathcal{T}$ is *monotonic*, if $s \rightarrow t$ implies $u[s\sigma]_p \rightarrow u[t\sigma]_p$ for all terms s, t, u in $\mathcal{T}$, substitutions σ, and positions p. An equational step $\leftrightarrow_E$, then, is the smallest monotonic extension of the symmetric closure of a system E of equations and $\leftrightarrow_E^*$ is the *provability* relation. The rewrite relation $\rightarrow_R$ is the smallest monotonic extension of a system R of rules and $\rightarrow_R^*$ is the *derivability* relation. Let $\succ$ be a monotonic, well-founded (strict) partial ordering on terms.[3] We call such an ordering a *reduction ordering*. A system R is terminating if, and only if, there exists a reduction ordering containing R. We refer to a rewrite system R as *confluent* if $\rightarrow_R$ is confluent; a terminating system will be called *convergent*. If, in addition, R presents the same theory as does E (i.e., $\leftrightarrow_R^* = \leftrightarrow_E^*$), then we say that R is *convergent* (or *complete*) for E. A *finite* convergent system R for E is a decision procedure for validity in the theory E, since an equation $s = t$ is valid in E (symbolized, $E \models s = t$ or $s =_E t$) if, and only if, it is provable in E (i.e., $E \models s = t$ or $s \leftrightarrow_E^* t$), which is the case, if, and only if, reducing s and t results in the identical term (i.e., $s \rightarrow_R^! v \leftarrow_R^! t$ for some v).

[3] *Well-founded* means that there is no endless descending chain $t_1 \succ t_2 \succ t_3 \succ \ldots$ of terms.

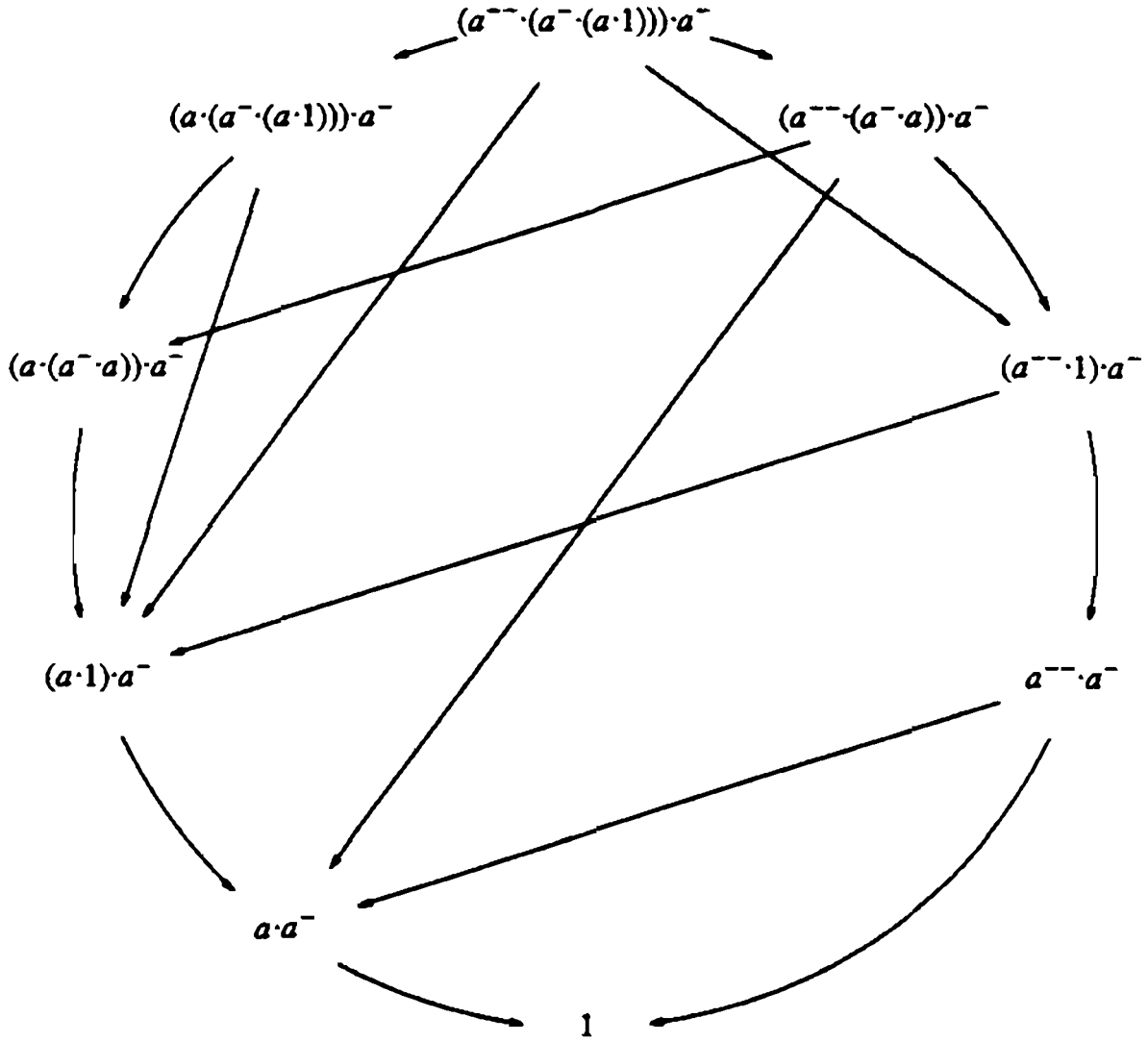


Fig. 1. A graph of derivations.

To summarize, a rewrite system R provides a decision procedure for an equational theory E, if the following four conditions hold:

a) R presents the same equational theory as E;
b) R is finite;
c) R is terminating;
d) R is confluent.

For example, the eight-rule rewrite system shown above decides validity in the three-axiom theory shown at the outset. To determine if two terms are always equal, the rules are repeatedly applied to both terms until normal forms are obtained. If the normal forms are identical, then, and only then, are the two terms equal in all models of the equations.

1.4. Completion

Given a finite set E of equations and a program for computing a reduction ordering $\succ$, the completion procedure (Knuth and Bendix, 1970) deduces consequences of E in its attempt to find a convergent system R for the theory presented by E. The central idea of completion is to limit attention to certain "critical" deductions obtained from overlappings of left-hand sides of rules. These overlappings are used to generate new rules, each of which is a *reduction* vis-a-vis $\succ$, i.e. $s \succ t$ whenever s rewrites to t, and each of which is *sound* for E, i.e., $s \leftrightarrow_E^* t$ whenever s rewrites to t. At the same time, rules are kept fully simplified: if $l \rightarrow r$ is a rule, then r is a normal form for the current system and l is not reducible by other rules. Such a fully-simplified system is said to be *reduced*. We will reserve the adjective *canonical* for a reduced convergent system. By reducing right-hand sides and deleting rules with rewritable left-hand sides, a convergent system can always be converted into a canonical one (see, e.g., Metivier (1983)).

RRL (Kapur and Sivakumar, 1983), REVE (Lescanne, 1983), FORMEL (Fages, 1984), RRLab (Thomas, 1984), and ERIL (Dick, 1986) are some current implementations of completion. Collections of canonical systems may be found in Butler and Lankford (1980), Hullot (1980a), and Le Chenadec (1985). Of course, not all equational theories are decidable, even if they can be presented by a finite set of equations (see, for instance, Davis (1958)). Nor can every decidable equational theory be decided via a convergent system (see, for instance, Kapur and Narendran (1985b)). But, when completion is successful, the resultant system can make for a very effective decision method. Given the above equational system and an ordering in which a term is greater than its proper subterms and 1 is minimal, the completion procedure will in fact generate our eight-rule decision procedure. Most of the afore-mentioned implementations provide help in choosing an ordering.[4]

In the next two sections we present an abstract version of completion and some of its extensions. We show how the method is used to generate decision procedures and how it is used as a semi-decision

[4] We will not usually bother to specify which reduction ordering is used; the reader can either take our word that an appropriate one exists, or consult Dershowitz (1987) for a survey of methods of proving termination.

procedure even when the process does not terminate. Section 4 (based on Dershowitz (1982b)) surveys applications of completion to

1) compute the congruence closure of sets of equations,
2) generate solutions of equations,
3) synthesize recursive programs,
4) prove theorems for inductively defined structures, and
5) prove theorems in first-order predicate calculus.

We conclude with brief mention of some further extensions.

A broad survey of the history of the ideas in completion and related procedures is given in Buchberger (1987). Surveys of rewriting theory include Huet and Oppen (1980), Benninghofen et al. (1987), Klop (1987), and Dershowitz and Jouannaud (1989).

2. The Completion Procedure

Given the axiomatization G

$1 \cdot x$	$\leftrightarrow$	x
$x^- \cdot x$	$\leftrightarrow$	1
$(x \cdot y) \cdot z$	$\leftrightarrow$	$x \cdot (y \cdot z)$

of group theory and an appropriate reduction ordering, the completion procedure generates the following canonical rewrite system:[5]

$1 \cdot x$	$\rightarrow$	x	$x \cdot 1$	$\rightarrow$	x
$x^- \cdot x$	$\rightarrow$	1	$x \cdot x^-$	$\rightarrow$	1
1^-	$\rightarrow$	1	x^{--}	$\rightarrow$	x
$y^- \cdot (y \cdot z)$	$\rightarrow$	z	$y \cdot (y^- \cdot z)$	$\rightarrow$	z
$(x \cdot y) \cdot z$	$\rightarrow$	$x \cdot (y \cdot z)$	$(x \cdot y)^-$	$\rightarrow$	$y^- \cdot x^-$

[5]This traditional axiomatization of abstract groups is the first example Knuth and Bendix tried out by hand, and the first experiment they ran their program on. It has become the "canonical" example of completion.

Three rules are oriented versions of the given axioms; the rest are the kind of lemmata one proves early on when learning group theory.[6] Actually, more are produced by the procedure, e.g., $x^{--} \cdot y \to x \cdot y$, but are subsequently simplified away. To prove, for example, that

$$(x^- \cdot (x \cdot y))^- = (x^- \cdot y)^- \cdot x^-$$

in group theory, both sides of the equation are reduced. Since

$$\begin{array}{l} (x^- \cdot y)^- \cdot x^- \\ \quad \to_R (y^- \cdot x^{--}) \cdot x^- \\ \quad\quad \to_R y^- \cdot (x^{--} \cdot x^-) \\ \quad\quad\quad \to_R y^- \cdot 1 \\ \quad\quad\quad\quad \to_R y^- \leftarrow_R (x^- \cdot (x \cdot y))^- \end{array}$$

the identity is valid. On the other hand, the two terms $(x^- \cdot y^-)^-$ and $(y^- \cdot x^-)^-$ reduce to the distinct terms $y \cdot x$ and $x \cdot y$, respectively; hence they are not equivalent.

2.1. Abstract Completion

Bachmair et al. (1986) have recently put completion in an abstract framework, an approach we adopt here. (See also Bachmair (1987) and Bachmair and Dershowitz (1989).) As in traditional proof theory (cf. Takeuti (1987)), proofs are reduced, in some well-founded sense, by replacing (locally) maximal subproofs with smaller ones, until a normal-form proof is obtained. In completion, the axioms used in proofs are in a constant state of flux; these changes are expressed as inference rules, which add a dynamic character to establishing the existence of reducible maximal subproofs. (See Dershowitz and Okada (1988).)

An *inference rule* (for our purposes) is a binary relation between pairs $(E; R)$, where E is a set of equations and R is a set of rewrite rules. Let $\succ$ be a reduction ordering and $\rhd$ a well-founded ordering on terms. We define the following set KB of six inference rules:

[6] As Knuth and Bendix wrote, "Without making use of any more ingenuity than can normally be expected of a computer's brain ..., [the first 65% of] the computation was done almost as a professional mathematician would have performed things."

Delete: $(E \cup \{s \leftrightarrow s\}; R) \vdash (E; R)$

Compose: $(E; R \cup \{s \to t\}) \vdash (E; R \cup \{s \to u\})$ if $t \to_R u$

Simplify: $(E \cup \{s \leftrightarrow t\}; R) \vdash (E \cup \{s \leftrightarrow u\}; R)$ if $t \to_R u$

Orient: $(E \cup \{s \leftrightarrow t\}; R) \vdash (E; R \cup \{s \to t\})$ if $s \succ t$

Collapse: $(E; R \cup \{s \to t\}) \vdash (E \cup \{u \leftrightarrow t\}; R)$
if $s \to_R u$ by a rule $l \to r \in R$ with $s \rhd l$

Deduce: $(E; R) \vdash (E \cup \{s \leftrightarrow t\}; R)$ if $s \leftarrow_R u \to_R t$

We write $(E; R) \vdash_{KB} (E'; R')$ if the latter may be obtained from the former by one application of a rule in KB.

(a) **Delete** removes a trivial equation $s \leftrightarrow s$. An equation $x \cdot 1 \leftrightarrow x \cdot 1$, for example, would be a candidate for deletion.

(b) **Compose** rewrites the right-hand side t of a rule $s \to t$, if possible. For example, given a rule $x^{--} \to x$, the rule $(x \cdot 1) \cdot 1 \to x^{--} \cdot 1$ would be replaced by $(x \cdot 1) \cdot 1 \to x \cdot 1$.

(c) **Simplify** rewrites either side of an equation $s \leftrightarrow t$. For example, given a rule $1 \cdot x \to x$, an equation $1 \cdot 1 \leftrightarrow 1$ would be replaced by $1 \leftrightarrow 1$.

(d) **Orient** turns an equation $s \leftrightarrow t$ that is orientable ($s \succ t$) into a rewrite rule. Since 1^- is greater than 1 for any reduction ordering (or an infinite derivation would be possible), the equation $1^- \leftrightarrow 1$ can only be oriented in the direction $1^- \to 1$.

(e) **Collapse** reduces the left-hand side of a rule $s \to t$ and turns the result into an equation $u \leftrightarrow t$, but only when the rule $l \to r$ being applied to s is smaller in some sense (embodied in $\rhd$) than the rule being removed. In practice, we use the (proper) *specialization* ordering as $\rhd$. In this ordering, $s \rhd l$ if a subterm of s is an instance of l (but not vice-versa). For example, a rule $x^{--} \cdot y \to x \cdot y$ collapses to $x \cdot y \leftrightarrow x \cdot y$ in the presence of a rule $x^{--} \to x$. The age of the two rules may also be taken into account when each left-hand side is an instance of the other, making older rules smaller.

(f) **Deduce** adds equational consequences to E, but only those that follow from back-to-back rewrites $s \leftarrow_R u$ and $u \to_R t$. For example, the rules $x \cdot x^- \to 1$ and $1 \cdot x \to x$ can both be applied to the term $1 \cdot 1^-$. The first rewrites this term to 1 and the second, to 1^-, from which the new equation $1^- \leftrightarrow 1$ can be deduced. As we will see, only consequences of certain "critical" peaks need to be considered.

We say that an equation is *simplifiable* if **simplify** can be applied to it, and that a rule is simplifiable if either **compose** or **collapse** applies.

It is not hard to see that the first five rules can only be applied a finite number of times (provided one starts out with finite E_0 and R_0, and with R_0 contained in $\succ$). Thus, new equational consequences should only be generated from existing rules after all rules and equations have been simplified as much as possible, and all trivial equations have been deleted. (In practice, these rules are usually best applied in the given order.)

A *completion procedure* is any program that takes (i) a finite set E_0 of equations, (ii) a finite set R_0 of rules, and (iii) a reduction ordering $\succ$ containing R_0, and uses the above rules to generate a sequence of inferences from $(E_0; R_0)$. Completion derives part of its power, as compared with "paramodulation" (Robinson and Wos, 1969), from the restriction of equational deduction to left-hand sides of rules only. It also saves space by preserving only fully simplified rules and equations. The *results* of a finite completion sequence $(E_0; R_0) \vdash_{KB} (E_1; R_1) \vdash_{KB} \cdots \vdash_{KB} (E_n; R_n)$ are E_n and R_n; in general, the results of a possibly infinite completion sequence $(E_0; R_0) \vdash_{KB} (E_1; R_1) \vdash_{KB} \cdots$ are the set $E^\infty = \cup_{i \geq 0} \cap_{j \geq i} E_j$ of persisting equations and the set $R^\infty = \cup_{i \geq 0} \cap_{j \geq i} R_j$ of persisting rules. We say that a completion sequence is *successful*, if E^∞ is empty and R^∞ is canonical. Table 1 presents a successful completion sequence for our first example. Beginning with three identities, the eight-rule system shown in the previous section is obtained.

The rules in KB are evidently *sound*, in that the class of provable theorems is unchanged by an inference step, i.e., $\leftrightarrow^*_{E \cup R} = \leftrightarrow^*_{E' \cup R'}$ whenever $(E; R) \vdash_{KB} (E'; R')$. Furthermore, as long as R is a subset of $\succ$, so is R', for which reason we require that $R_0 \subseteq \succ$. We are thus assured that the result R^∞ of any (finite or infinite) successful completion sequence is terminating and presents the same equational theory as did $E_0 \cup R_0$ (with R_0 considered as equations).

We are particularly interested in the degree of confidence one can have in the eventual success of specific completion procedures. We will say that a completion (or similar) procedure is *correct* if R^∞ is canonical whenever E^∞ is empty, regardless of whether the completion sequence is infinite or finite. It is in this sense that Huet (1981) proved the Knuth–Bendix procedure correct.

i	R_i	E_i	inference
0		$1 \cdot x \leftrightarrow x$ $x \cdot 1 \leftrightarrow x$ $y^- \cdot (y \cdot z) \leftrightarrow z$	
1	$x \cdot 1 \rightarrow x$	$1 \cdot x \leftrightarrow x$ $y^- \cdot (y \cdot z) \leftrightarrow x$	orient
2	R_1 $1 \cdot x \rightarrow x$	$y^- \cdot (y \cdot z) \leftrightarrow z$	orient
3	R_2 $y^- \cdot (y \cdot z) \rightarrow z$		orient
4		$y^- \cdot y \leftrightarrow 1$	deduce (1,3)
5	R_3 $y^- \cdot y \rightarrow 1$		orient
6		$1^- \leftrightarrow 1$	deduce (1,5)
7	R_4 $1^- \rightarrow 1$		orient
8		$y^{--} \cdot z \leftrightarrow y \cdot z$	deduce (3,3)
9	R_7 $y^{--} \cdot z \rightarrow y \cdot z$		orient
10		$1^- \cdot x \leftrightarrow 1 \cdot x$	deduce (7,9)
11	R_8 $1^- \cdot x \rightarrow 1 \cdot x$		orient
12	R_9 $1^- \cdot z \rightarrow z$		compose (11,2)
13		$1 \cdot x \leftrightarrow x$	collapse (12,7)
14		$x \leftrightarrow x$	simplify (2)
15	R_0		delete
16		$y^{--} \leftrightarrow y$	deduce (1,9)
17	R_6 $y^{--} \rightarrow y$		orient
18		$y \cdot z \leftrightarrow y \cdot z$	collapse (9,17)
19			delete
20		$y \cdot (y^- \cdot x) \leftrightarrow x$	deduce (3,17)
21	R_{17} $y \cdot (y^- \cdot z) \rightarrow z$		orient
22		$y \cdot y^- \leftrightarrow 1$	deduce (1,21)
23	R_{21} $y \cdot y^- \rightarrow 1$		orient

Table 1. A successful completion sequence for a fragment of group theory.

Ascertaining correctness is complicated by the simplification of rules allowed by the inference system. We will say that a procedure is *complete* if E^∞ is always empty and R^∞ is always canonical; for completeness, a stronger inference system is generally required. (Correctness and completeness can be weakened to refer only to convergence. not canonicity, as in Bachmair (1987).)

2.2. Proof Simplification

Each proof step $t_i \leftrightarrow_{E\cup R} t_{i+1}$ in a proof $t_1 \leftrightarrow_{E\cup R} t_2 \leftrightarrow_{E\cup R} \ldots \leftrightarrow_{E\cup R} t_n$, is either an equational step $t_i \leftrightarrow_E t_{i+1}$, a rewrite step $t_i \rightarrow_R t_{i+1}$, or a backwards rewrite step $t_i \leftarrow_R t_{i+1}$. In any case, each step must be justified by specifying the equation (in E) or rule (in R) being used, as well as the position at which it is being applied (and the substitution used). By applying the inference rules in KB, inferring $(E'; R')$ from $(E; R)$, it may be possible to simplify a given proof in $E\cup R$, replacing some proof steps with alternate ones from $E'\cup R'$.

In general, our goal is to transform proofs into *normalized* ones by completely eliminating certain patterns. To formalize this idea, we define a relation $\Rightarrow_{KB}$ between proofs. Let E be a set of equations closed under equational deduction and R be the subset of E contained in a reduction ordering $\succ$. By a *proof pattern* we mean a schema describing a class of subproofs in $(E; R)$; e.g., to characterize rewrite proofs in R, we use the pattern $s \rightarrow_R^* v \leftarrow_R^* t$, where s, t, and v denote arbitrary terms. We refer to the pattern $s \leftarrow_R u \rightarrow_R t$ as a *peak*. If a proof contains neither an equational step nor a peak, it must be a rewrite proof. The proof relation $\Rightarrow_{KB}$ is defined by the following set of pairs of patterns—which we will also refer to as KB:

$$
\begin{array}{rcl}
s \leftrightarrow_E t & \Rightarrow & s \rightarrow_R^* v \leftarrow_R^* t \\
s \leftrightarrow_E t & \Rightarrow & s \rightarrow_R^* u \leftrightarrow_E v \leftarrow_R^+ t \\
s \leftarrow_R w \rightarrow_R t & \Rightarrow & s \rightarrow_R^* v \leftarrow_R^* t \\
s \leftarrow_R w \rightarrow_R t & \Rightarrow & s \rightarrow_R^* u \leftrightarrow_E v \leftarrow_R^* t \\
s \rightarrow_R t & \Rightarrow & s \rightarrow_R u \leftarrow_R^+ t \\
\multicolumn{3}{r}{\text{where } s \rightarrow_R t \text{ by } l \rightarrow r \text{ and } s \rightarrow_R u \text{ by } g \rightarrow d \text{ and } l = g} \\
s \rightarrow_R t & \Rightarrow & s \rightarrow_R u \leftrightarrow_E v \leftarrow_R^* t \\
\multicolumn{3}{r}{\text{where } s \rightarrow_R t \text{ by } l \rightarrow r \text{ and } s \rightarrow_R u \text{ by } g \rightarrow d \text{ and } l \rhd g}
\end{array}
$$

An equational step may be replaced by a (possibly empty) rewrite proof or by a rewrite proof with a single simpler equational step at the bottom of the "valley"; a peak may be replaced by a rewrite proof, possibly containing an equational step at the bottom; and a single rewrite step may be replaced by a simpler rewrite proof, possibly containing an equational step at the bottom. We do not explicitly distinguish between a proof $t_1 \leftrightarrow \ldots \leftrightarrow t_n$ and its inversion $t_n \leftrightarrow \ldots \leftrightarrow t_1$. For example, the last two rules transform the symmetric step $t \leftarrow_R s$, too, giving a smaller proof $t \leftrightarrow_{R \cup E} s$.

This proof rewriting system eliminates equational steps, peaks, and applications of unsimplified rules from proofs, whenever possible. Furthermore:

Proposition 1. *The proof relation $\Rightarrow_{KB}$ is terminating (Bachmair et al. (1986)).*

That is, there can be no infinite derivation $P_1 \Rightarrow_{KB} P_2 \Rightarrow_{KB} \ldots$ of proofs P_i in $E \cup R$ (R itself is terminating). Thus, $\Rightarrow_{KB}$ encompasses a well-defined notion of "simplifying" proofs. It can be applied until further simplification is impossible; its normal forms are rewrite proofs and proofs containing equations, the two sides of which are incomparable under $\succ$.

Proof. Consider the partial ordering $>_{kb}$ that compares proofs by comparing multisets of proof steps, where the cost of a proof step is measured according to the triple $(\{s\}, l, t)$ for each rewrite step $s \rightarrow_R t$ that is an application of a rule $l \rightarrow r$, and according to $(\{s, t\}, l, t)$ for each application $s \leftrightarrow_E t$ of an equation $l \leftrightarrow r$. Triples are compared lexicographically, using the multiset ordering $\succ\!\!\succ$ induced by the given ordering $\succ$ for the first component, $\rhd$ for the second, and $\succ$ for the third, and multisets of triples are compared in the induced multiset ordering (as defined in Dershowitz and Manna (1979)). Assuming that $\succ$ is well-founded, this ordering $>_{kb}$ is also well-founded. It is also monotonic on proofs, i.e., if $P >_{kb} Q$ for two proofs P and Q of the same equation, then any proof that contains an instance of P as a subproof is greater under $>_{kb}$ than a similar proof containing an instance of Q. Since the rules KB are contained in the monotonic well-founded ordering $>_{kb}$, the proof relation $\Rightarrow_{KB}$ is terminating. □

The inference system KB and proof relation KB are related by the following lemma:

Reflection Lemma. *If* $(E; R) \vdash_{KB} (E'; R')$, *then for every proof* P *in* $E \cup R$ *there exists a proof* P' *in* $E' \cup R'$ *such that* $P \Rightarrow^*_{KB} P'$ *(Bachmair (1987)).*

In this way, the inference system $\vdash_{KB}$ is used to generate rules needed for $\Rightarrow_{KB}$ to be applicable. It is not enough, however, to know that inferences do not make more complex proofs necessary; we need to establish that applying inference rules to a non-rewrite proof will actually lead to a strictly simpler proof vis-a-vis $\Rightarrow_{KB}$. Since the latter is terminating, this means that a rewrite proof will eventually be attained. As we will see, this is only the case when certain "fairness" conditions are met.

This view of completion as progressively rewriting proofs is advocated in Bachmair et al. (1986), Bachmair (1987), and Bachmair and Dershowitz (1988) and was expounded on by Jouannaud (1987); the idea of establishing confluence by normalizing equational proofs appears also in Küchlin (1986b); the use of multiset orderings in such a context was pioneered by Jouannaud and Kirchner (1984).

2.3. Critical Pairs

Since the rule **deduce** can lead to infinitely long chains of inference, fairness conditions aim to minimize applications of that rule, while ensuring that it is not completely ignored. Let $l \to r$ and $g \to d$ be two rules. We say that g *overlaps* l if there is a *nonvariable* subterm s at a position p of l and a (unifying) substitution σ for the variables of the two rules, such that $s\sigma = l\sigma/p = g\sigma$. In that case, the overlapped term $l\sigma$ can be rewritten to either $r\sigma$ or $l\sigma[d\sigma]_p$. The variables of the overlapping rules are considered disjoint, even if the two rules are actually one and the same. (Since we are presuming termination, no left-hand side is just a variable.)

Definition 1. The equation $d\mu \leftrightarrow g\mu[r\mu]_p$ is a *critical pair* of two rules $l \to r$ and $g \to d$ (whose variables have been renamed, if necessary, so that the rules have none in common), if there exists a *most general* unifier μ of g/p and l for some nonvariable position p in g (Knuth and Bendix (1970)).

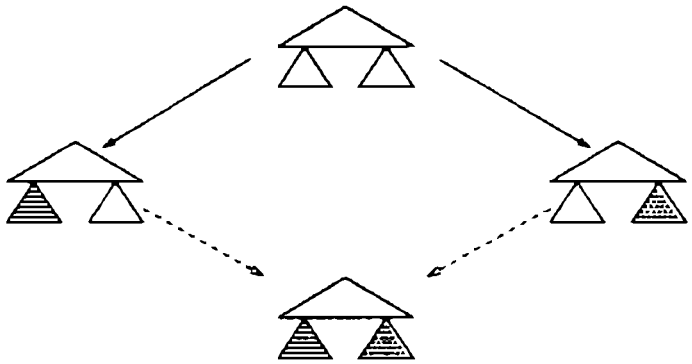
For example, $x^{-}{}^{-}{}^{-} \cdot (x \cdot y) \leftrightarrow y$ is a critical pair obtained by overlapping $x^{-}{}^{-} \cdot y \to x \cdot y$ at the subterm $x \cdot y$ in the left-hand side of $x^{-} \cdot (x \cdot y) \to y$.

Let $\mathbf{cp}(R)$ denote the set of all critical pairs between (not necessarily distinct) rules in R and $\leftrightarrow_{\mathbf{cp}(R)}$, the monotonic extension of those equations.

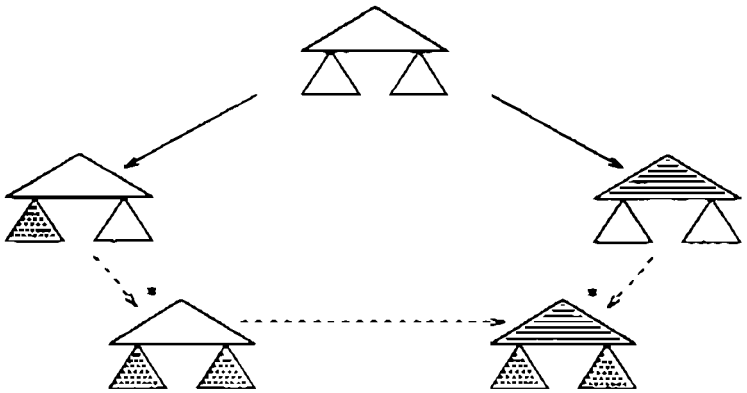
Critical Pair Lemma. *For any rewrite system R and peak $s \leftarrow_R u \rightarrow_R t$, there either exists a rewrite proof $s \rightarrow_R^* v \leftarrow_R^* t$ or a critical-pair proof $s \leftrightarrow_{\mathbf{cp}(R)} t$ (Knuth and Bendix (1970)).*

In set notation: $\leftarrow_R \circ \rightarrow_R \subseteq (\rightarrow_R^* \circ \leftarrow_R^*) \cup \leftrightarrow_{\mathbf{cp}(R)}$.

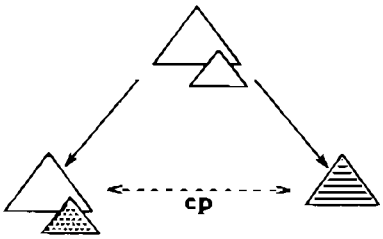
Proof. Behold Fig. 2! □



(a) Disjoint case.



(b) Variable overlap case.



(c) Critical overlap case.

Fig. 2. Proof of Critical Pair Lemma by cases.

It follows from this and Newman's Lemma (Newman (1942); see Huet (1981)) that a terminating system R is confluent if, and only if, $\mathbf{cp}(R)$, viewed as a relation, is contained in $\rightarrow_R^* \circ \leftarrow_R^*$ (Knuth and Bendix (1970)).

Definition 2. A completion sequence in KB is *fair* if all persistent critical pairs are generated ($\mathbf{cp}(R^\infty)$ is a subset of $\cup E_i$), no

simplifiable rule persists (R^∞ is reduced), and no equation persists (E^∞ is empty).

Rules or equations that differ only in the names of their variables are, for all intents and purposes, treated as identical. Table 1 only contains deductions from critical pairs; Table 2 extends it to a point when it becomes fair, by checking all remaining critical pairs. (Trivial ones, obtained by superposing a left-hand side on all of itself, are omitted from the tables.) In a more abstract framework, one could define a completion sequence to be fair with respect to a given set of proof patterns, if for any proof containing an instance of one of those patterns, there exists a (later) step at which a strictly simpler proof is possible. See Bachmair (1987).

For any given completion sequence $(E_0; R_0) \vdash_{KB} (E_1; R_1) \vdash_{KB} \ldots$, we let $\leftrightarrow_i$ stand for $\leftrightarrow_{E_i \cup R_i}$, i.e., for a proof step in either E_i or R_i.

Proof Normalization Theorem. *If a completion sequence* $(E_0; R_0) \vdash_{KB} (E_1; R_1) \vdash_{KB} \ldots$ *is fair, then for any proof* $s \leftrightarrow_i^* t$ *in* $E_i \cup R_i$, *there is a rewrite proof* $s \rightarrow_{R^\infty}^! v \leftarrow_{R^\infty}^! t$ *in* R^∞ *(Huet (1981)).*

Huet (1981) proves this for a specific completion procedure; the following proof is more general:

Proof. The proof is by induction with respect to the well-founded ordering $\Rightarrow_{KB}^+$. Suppose that $s \leftrightarrow_i^* t$ is not a persistent rewrite proof $s \rightarrow_{R^\infty}^* v \leftarrow_{R^\infty}^* t$. Then it must be reducible by $\Rightarrow_{KB}$, either on account of a peak (using the Critical Pair Lemma) or the application of a nonpersisting step. Thus, $s \leftrightarrow_i^* t \Rightarrow_{KB}^+ s \leftrightarrow_j^* t$ for some step j and by induction $s \rightarrow_{R^\infty}^* w \leftarrow_{R^\infty}^* t$ for some w. Since R^∞ is normalizing, we have $w \rightarrow_{R^\infty}^! v$ for some v. (Bachmair et al., (1986)). □

Thus, an n-step completion sequence *succeeds* if it is fair, i.e., if E_n is empty, R_n is reduced, and each of the latter's critical pairs already appeared in some E_i. We say that it *fails* at step n if no fair sequence has, as a prefix, the sequence generated up to that point; in that case there is little point continuing. Thus, a completion procedure may be said to be *fair* if it is successful whenever possible, i.e., if it only generates fair or failing sequences. Assuming the procedure never

i	R_i	E_i	inferences
24		$1 \leftrightarrow 1$	deduce (1,2)
25			delete
26		$1^{-} \cdot (1 \cdot x) \leftrightarrow x$	deduce (2,3)
27		$1 \cdot (1 \cdot x) \leftrightarrow x$	simplify (7)
28–29	$x \cdot 1 \underset{1}{\rightarrow} x$	$x \leftrightarrow x$	simplify (2,2)
30			delete
31	$1 \cdot x \underset{2}{\rightarrow} x$	$y^{--} \cdot 1 \leftrightarrow y$	deduce (3,5)
32–33		$y \leftrightarrow y$	simplify (17,1)
34	$y^{-} \cdot (y \cdot z) \underset{3}{\rightarrow} z$		delete
35		$1 \cdot 1 \leftrightarrow 1$	deduce (5,7)
36	$y^{-} \cdot y \underset{5}{\rightarrow} 1$	$1 \leftrightarrow 1$	simplify (1)
37			delete
38	$1^{-} \underset{7}{\rightarrow} 1$	$y^{-} \leftrightarrow y^{-}$	deduce (17,17)
39			delete
40	$y^{--} \underset{17}{\rightarrow} y$	$1^{-} \cdot z \leftrightarrow z$	deduce (2,21)
41–42		$z \leftrightarrow z$	simplify (7,2)
43	$y \cdot (y^{-} \cdot z) \underset{21}{\rightarrow} z$		delete
44		$y^{-} \cdot z \leftrightarrow y^{-} \cdot z$	deduce (3,21)
45	$y \cdot y^{-} \underset{23}{\rightarrow} 1$		delete
46		$y^{-} \leftrightarrow y^{-} \cdot 1$	deduce (3,23)
47		$y^{-} \leftrightarrow y^{-}$	simplify (1)
48			delete

Table 2. A fair continuation of Table 1.

discriminates against any critical pair or simplifiable rule or equation, the only possible reason for failure is an unorientable equation.

Practically speaking, what this means is that the **deduce** rule is restricted to generating critical pairs, of which there can only be

finitely many at any given time. Each time a new rule is generated (by **orient**), it can give rise to new critical pairs formed with the old rules or with itself. All those pairs need eventually to be simplified or oriented, unless the new rule itself is later simplified, in which case it becomes no longer necessary to consider them. A marking scheme is generally used to keep track of which rules still need to be overlapped with which; see, for instance, Huet (1981).

For any given equational theory, there can be only one (finite or infinite) canonical rewrite system whose rewrite relation is contained in a given reduction ordering (Butler and Lankford (1980), Lankford and Ballantyne (1983), and Metivier (1983)). This uniqueness result is up to renaming of variables and depends on the systems being reduced. Thus, two successful derivations (finite or infinite), given the same inputs, must result in the identical canonical system. It follows that, if there exists a finite canonical R contained in a reduction ordering $\succ$ and whose equational theory is E_0, then a fair procedure (given $\succ$ and starting with E_0) can either succeed—providing a decision procedure $R_n = R$ for validity in the theory— or fail (on account of unorientable equations), but cannot have only infinite fair sequences (Dershowitz et al. (1988)). The procedures given in Knuth and Bendix (1970) and Huet (1981)—though correct—are not fair in our sense, since they sometimes abort upon generating an unorientable critical pair, even if choosing a different pair might result in success. When they do not abort, they are fair, and find a canonical system if one exists.[7] A completion procedure need not abort, however, unless *all* critical pairs are unorientable and unsimplifiable (as, for instance, is the case with the inefficient procedure given in Dershowitz (1982b)).

2.4. Semi-Deciding Validity

We have seen how completion generates a decision procedure whenever it succeeds after a finite number of steps. Completion is also applicable to theorem-proving tasks when it does not terminate with a convergent system. By the Proof Normalization Theorem, the

[7] It is somewhat unfair to categorize Huet's procedure as unfair, since it was Huet who introduced (a weaker notion of) "fairness" in the context of completion, allowing, however, for unnecessary abortion.

(possibly infinite) result R^∞ of a fair completion sequence must be canonical (as first shown by Huet (1981)). Since, by soundness, $\leftrightarrow^*_\infty = \leftrightarrow^*_0$, completion provides a semi-decision procedure for validity in the given theory—when it does not fail. That is, if $s =_{E_0 \cup R_0} t$, then, for any unfailing sequence, there is some step n when it is possible to ascertain that $s \rightarrow^!_{R_n} v \leftarrow^!_{R_n} t$ for some v. For example, given the decidable theory[8]

$$\boxed{\begin{array}{rcl} (x \cdot y) \cdot z & \leftrightarrow & x \cdot (y \cdot z) \\ (x \cdot x) & \leftrightarrow & x \end{array}}$$

of *bands* (idempotent semigroups), and a straightforward ordering, completion generates an infinite set R^∞ of rules, including:

$$\begin{array}{rcl} (x \cdot y) \cdot z & \rightarrow & x \cdot (y \cdot z) \\ (x \cdot x) & \rightarrow & x \\ x \cdot (x \cdot z) & \rightarrow & x \cdot z \\ x \cdot (y \cdot (x \cdot y)) & \rightarrow & x \cdot y \\ x \cdot (y \cdot (x \cdot (y \cdot z))) & \rightarrow & x \cdot (y \cdot z) \\ \vdots & & \\ x \cdot (y \cdot (z \cdot (y \cdot (x \cdot (y \cdot (z \cdot x))))))) & \rightarrow & x \cdot (y \cdot (z \cdot x)) \\ \vdots & & \end{array}$$

To prove that

$$((x \cdot ((y \cdot z) \cdot y)) \cdot x) \cdot ((y \cdot z) \cdot x) = (x \cdot y) \cdot (y \cdot ((x \cdot y) \cdot (z \cdot x)))$$

in this theory, the two sides are reduced by the partial system shown above. Since they both reduce to $x \cdot (y \cdot (z \cdot x))$, the equation is valid. But, since R^∞ is infinite, it does not constitute a decision procedure.

The use of completion as an equational theorem prover can be rephrased in a refutational framework, as in Lankford (1975):

Corollary 1. *Let s and t be two terms and s' and t' their Skolemized versions (i.e., with their variables treated as constants). Let R^∞ be the result of a fair completion sequence starting from*

[8] See Siekmann and Szabó (1982).

$(E; \{eq(x,x) \to T, eq(s',t') \to F\})$, where eq is a new binary symbol, T and F are new constants, and any equality term $eq(u,v)$ is greater (under the given reduction ordering) than F which is greater than T. Then $s =_E t$ if, and only if, R^∞ contains the contradiction $F \to T$.

Completion is, however, an incomplete theorem proving method, on account of its potential for failure (but see Section 3.2). Note that the rules $eq(x,x) \to T$ and $eq(s',t') \to F$ in the initial set play a very limited role in the completion process. The first persists until the very end; the second gets repeatedly collapsed into new variable-free rules of the form $eq(u,v) \to F$. Critical pairs involving $eq(u,v) \to F$ are either trivial (if both rules are of this form), superfluous (if they can be generated by collapsing u or v), or lead directly to a contradiction (when u and v are identical).

3. Extensions

In this section we touch upon various difficulties that may befall completion and indicate some proposed solutions.

3.1. Nontermination

The outcome of completion strongly depends on the choice of reduction ordering used to orient equations (see, for example, Lescanne (1986) or Gnaedig (1987)). Though implementations of completion typically provide the user with help in finding an ordering (see, in particular, Detlefs and Forgaard (1985) and Martin (1987)), choosing one that leads to success after a finite number of steps remains problematic. For example, the orientation of the first equation makes the difference between finding or not finding a finite convergent system for the following theory E_0:[9]

$(x \cdot y)^-$	$\leftrightarrow$	$y^- \cdot x^-$
$(x \cdot y) \cdot z$	$\leftrightarrow$	$x \cdot (y \cdot z)$

[9] See Lescanne (1984).

On the one hand, we have the finite successful completion sequence

$$\begin{aligned}(E_0;\emptyset) &\vdash_{KB} (\{(x\cdot y)\cdot z \leftrightarrow x\cdot(y\cdot z)\};\{(x\cdot y)^- \to y^-\cdot x^-\})\\ &\vdash_{KB} (\emptyset;\{(x\cdot y)\cdot z \to x\cdot(y\cdot z),(x\cdot y)^- \to y^-\cdot x^-\});\end{aligned}$$

on the other, there is an infinite (albeit successful) one:

$$\begin{aligned}(E_0;\emptyset) &\vdash_{KB} (\{(x\cdot y)\cdot z \leftrightarrow x\cdot(y\cdot z)\};\{y^-\cdot x^- \to (x\cdot y)^-\})\\ &\vdash_{KB} (\emptyset;\{(x\cdot y)\cdot z \to x\cdot(y\cdot z),y^-\cdot x^- \to (x\cdot y)^-\})\\ &\vdash_{KB} (\{(x\cdot y)^-\cdot z \leftrightarrow x^-\cdot(y^-\cdot z)\};\\ &\qquad \{(x\cdot y)\cdot z \to x\cdot(y\cdot z),\ y^-\cdot x^- \to (x\cdot y)^-\})\\ &\vdash_{KB} (\emptyset;\{(x\cdot y)^-\cdot z \to x^-\cdot(y^-\cdot z),\\ &\qquad (x\cdot y)\cdot z \to x\cdot(y\cdot z),y^-\cdot x^- \to (x\cdot y)^-\})\\ &\vdash^+_{KB} (\emptyset;\{(x\cdot y)^{--}\cdot\ z \to x^{--}\cdot(y^{--}\cdot z),\\ &\qquad (x\cdot y)^-\cdot z \to x^-\cdot(y^-\cdot z)\},\\ &\qquad (x\cdot y)\cdot z \to x\cdot(y\cdot z),y^-\cdot x^- \to (x\cdot y)^-\})\\ &\vdash_{KB} \ldots\end{aligned}$$

Indeed, the class of orderings used in the above-mentioned implementations is insufficient for proving termination of all terminating systems. For the purposes of generating a canonical system (as opposed to the use of completion as a theorem prover), one can delay testing for termination until all critical pairs have been considered. In this approach, an equation is oriented into a new rule, as long as the enlarged system is not known to be nonterminating. Of course, we are no longer assured that the result is terminating and must be careful not to simplify an equation or rule without limit. Along these lines, Plaisted (1986) suggests using the necessary but insufficient condition for nontermination that R' be homeomorphically self-embedding, whereas Purdom (1987) suggests using the sufficient but unnecessary condition that R' be obviously looping. (See Plaisted (1985a) for the definitions of "self-embedding" and "looping" and proofs that they are, in general, undecidable properties of rewrite systems.)

3.2. Unfailing Completion

It is possible for some completion sequences to fail while others succeed (Avenhaus, 1985; Dershowitz et al., 1988). For example, let $\succ$

be a reduction ordering in which k is greater than m and n, which are themselves incomparable, but are both greater than c, and terms are greater than their proper subterms. We have a successful sequence:

$$\begin{aligned}(\{k \leftrightarrow m, k \leftrightarrow n, f(k) \leftrightarrow c\}; \{f(m) \to m\}) \\ &\vdash_{KB} (\{k \leftrightarrow n, f(k) \leftrightarrow c\};\ \{k \to m, f(m) \to m\}) \\ &\vdash^{+}_{KB} (\{m \leftrightarrow n, m \leftrightarrow c\};\ \{k \to m, f(m) \to m\}) \\ &\vdash_{KB} (\{m \leftrightarrow n\};\ \{m \to c, k \to m, f(m) \to m\}) \\ &\vdash^{+}_{KB} (\{f(c) \leftrightarrow c, c \leftrightarrow n\};\ \{m \to c, k \to c\}) \\ &\vdash^{+}_{KB} (\emptyset;\ \{f(c) \to c, n \to c, m \to c, k \to c\}),\end{aligned}$$

as well as a failing one:

$$\begin{aligned}(\{k \leftrightarrow m, k \leftrightarrow n, f(k) \leftrightarrow c\}; \{f(m) \to m\}) \\ &\vdash_{KB} (\{k \leftrightarrow m, f(k) \leftrightarrow c\};\ \{k \to n, f(m) \to m\}) \\ &\vdash^{+}_{KB} (\{n \leftrightarrow m\};\ \{f(n) \to c, k \to n, f(m) \to m\}).\end{aligned}$$

Thus, it may be advisable to implement completion with backtracking to explore alternative sequences upon failure.

For a given reduction ordering $\succ$ and equational theory E, there exists a (not necessarily finite) convergent system for E contained in $\succ$ if, and only if, each congruence class of terms equal under E has a unique minimal element vis-a-vis $\succ$ (Avenhaus, 1985). Unfortunately, even when such a system exists, it may be the case that the zero-step sequence $(E;\emptyset)$ is already a failure—and backtracking would be to no avail (Dershowitz et al., 1988). For example, with the same ordering as above, no fair sequence exists for $E_0 = \{f(n) \leftrightarrow c, f(m) \leftrightarrow m, m \leftrightarrow n\}$, despite the existence of $R = \{f(c) \to c, m \to c, n \to c\}$. Note that a stronger ordering (one that makes m and n comparable) would lead to success in this case, but that this too is not always possible (replace m and n by the inherently incomparable terms, $g(x)$ and $h(y)$, respectively). Knuth and Bendix (1970) suggest circumventing such failures by introducing a new minimal function symbol (in the above case, a constant, say k) and adding new rules equating the problematic terms to a term headed by the new symbol (in our case, $m \to k$ and $n \to k$) and having the terms' variables as its immediate subterms. The resulting rewrite system $(\{m \to k, n \to k, c \to k, f(k) \to k\})$

would no longer represent the same equational theory as the original axioms (it is, rather, a conservative extension thereof), but could still be used as a decision procedure for the latter. This approach, however, often degenerates into an infinite succession of new symbols, particularly in the presence of "permuters," identities whose sides differ only in the location of variables. For example, the equation $x \cdot y \leftrightarrow y \cdot x$ cannot be oriented by any reduction ordering, and replacing it by the rules $x \cdot y \rightarrow k(x, y)$ and $y \cdot x \rightarrow k(x, y)$ just leads to the equally problematic critical pair $k(x, y) \leftrightarrow k(y, x)$.

Were the reduction ordering $\succ$ on $\mathcal{T}$ that is supplied to the procedure *total* (or at least total on each class of equivalent terms), then all equations would be orientable, and failure could be avoided. Unfortunately, as we have just seen, that cannot, in general, be the case. On the other hand, it is always possible to define a total reduction ordering on variable-free terms. Taking advantage of the existence of such an ordering, one can overcome the problem of unorientable equations (containing variables). This is the "unfailing" method described in Hsiang and Rusinowitch (1987) and Bachmair et al. (1989), which is based on the early, more general methods suggested by Brown (1975), Lankford (1975), and Plaisted (personal communication). The older methods treat unorientable equations $s \leftrightarrow t$ as two rules, $s \rightarrow t$ and $t \rightarrow s$, for the purposes of critical pair generation (like paramodulation), but refrain from their careless use for simplifying equations or rules. The newer methods treat such an equation as (perhaps infinitely) many oriented instances by incorporating inferences:

Paramodulate: $(E; R) \vdash (E \cup \{s \leftrightarrow t\}; R)$
if $s \leftrightarrow_E u \leftrightarrow_E t$ and $s, t \not\succeq u$

With this rule, an equation $s \leftrightarrow t$ is inferred only if it results from a subproof $s \leftrightarrow_E u \leftrightarrow_E t$, some instances of which may be a peak. Only critical subproofs, as defined by appropriate fairness conditions, need to be generated. Under reasonable assumptions, the unfailing completion procedure is complete. See Bachmair et al. (1989) for a treatment within the inference-rule framework.

3.3. Associative-Commutative Completion

To handle some common problematic identities, such as commutativity (with or without associativity), reasonably efficiently, special-purpose completion procedures have been designed. Let A be an equational system; for any rewrite system R, we define a rewrite relation $\rightarrow_{R/A}$ on A-congruence classes. Thus, we write $s \rightarrow_{R/A} t$ if $s \leftrightarrow_A^* u \rightarrow_R v \leftrightarrow_A^* t$ for some terms u and v. We are particularly interested in the AC equational system, consisting of two axioms:

$$\begin{array}{|c|} \hline f(x,y) \leftrightarrow f(y,x) \\ f(f(x,y),z) \leftrightarrow f(x,f(y,z)) \\ \hline \end{array}$$

for each function symbol f in some subset of the binary symbols in $\mathcal{F}$. So as to limit deductions to "critical" ones, the definition of overlap is extended to include cases in which two rules can be applied near the top of the same term (Lankford and Ballantyne, 1977b). Functions that are only commutative can be handled similarly (Lankford and Ballantyne, 1977a).

Let $\succ$ be a reduction ordering and $\rhd$ a well-founded ordering on terms, both of which are compatible with an equational system A. By *compatibility* with A, we mean that $u \leftrightarrow_A^* s \succ t \leftrightarrow_A^* v$ implies $u \succ v$ and $u \leftrightarrow_A^* s \rhd t \leftrightarrow_A^* v$ implies $u \rhd v$. We define the following set KB/A of inference rules:

Delete:	$(E \cup \{s \leftrightarrow t\}; R) \vdash (E; R)$	if $s \leftrightarrow_A^* t$
Compose:	$(E; R \cup \{s \rightarrow t\}) \vdash (E; R \cup \{s \rightarrow u\})$	if $t \rightarrow_{R/A} u$
Simplify:	$(E \cup \{s \leftrightarrow t\}; R) \vdash (E \cup \{s \leftrightarrow u\}; R)$	if $t \rightarrow_{R/A} u$
Orient:	$(E \cup \{s \leftrightarrow t\}; R) \vdash (E; R \cup \{s \rightarrow t\})$	if $s \succ t$
Collapse:	$(E; R \cup \{s \rightarrow t\}) \vdash (E \cup \{u \leftrightarrow t\}; R)$	if $\rightarrow_{R/A} u$ by a rule $l \rightarrow r \in R$ with $s \rhd l$
Extend:	$(E; R) \vdash (E; R \cup \{t \rightarrow s\})$	if $s \leftarrow_R u \leftrightarrow_A t$
Deduce:	$(E; R) \vdash (E \cup \{s \leftrightarrow t\}; R)$	if $s \leftarrow_R u \rightarrow_{R/A} t$

(a) **Delete** removes an equation between terms that are equal in A.

(b–d) **Compose, simplify**, and **orient**, are analogous to their counterparts in KB.

(e) **Collapse** simplifies left-hand sides according to the specialization ordering $\rhd$ "modulo" A; that is, $s \rhd l$ if a (not necessarily proper) subterm of s is equivalent under A to an instance of l, but not vice-versa. We also require that all the A steps used in applying $l \to r$ to s are strictly *below* the top of s.

(f) **Extend** adds consequences of R resulting from A steps. For example, an AC-extended version of $x \cdot x \to x$ is $x \cdot (x \cdot z) \to x \cdot z$.

(g) **Deduce** adds equational consequences to E, including those that require some A steps. For example, from $x \cdot x^{-} \to 1$ and $x \cdot 1 \to x$, one can deduce $1^{-} \leftrightarrow 1$, since $1 \cdot 1^{-} \leftrightarrow_{AC} 1^{-} \cdot 1$.

We write $(E; R) \vdash_{KB/A} (E'; R')$ if the latter may be obtained from the former by one application of a rule in KB/A. When A is empty, this inference system is identical with that for standard completion. As with standard completion, we wish to limit application of the inference rules, **extend** and **deduce**, to "critical" situations, at the same time ensuring that nonrewrite proofs will eventually be simplified.

Peterson and Stickel (1981) introduced a new relation $\to_{A\backslash R}$, which is weaker than $\to_{R/A}$. It may be defined by the set of rules

$$A\backslash R = \{s \to r : s \leftrightarrow_A^* l \text{ and } l \to r \in R\}$$

which includes (perhaps infinitely many) A-variants of left-hand sides of rules in R. A rule in $A\backslash R$ is applied not only to exact instances of an s, but also when the subterms matching different instances of the same variable in s are equal in A. Thus, $s \to_{A\backslash R} s[r\sigma]_p$ if $s/p \leftrightarrow_A^* l\sigma$ for some position p in s and rule $l \to r$ in R. Computing this relation for AC requires associative-commutative pattern matching (Stickel, 1981). For example, if $x \cdot x \to x$ is a rule in R and $\cdot$ is associative and commutative, then $(a \cdot b) \cdot (b \cdot a) \to_{AC\backslash R} a \cdot b$ and $(a \cdot (b \cdot b)) \cdot a \to_{R/AC} a \cdot (b \cdot b)$, but $(a \cdot (b \cdot b)) \cdot a \not\to_{AC\backslash R} a \cdot (b \cdot b)$. The notions of "irreducible term" and "reduced systems" may be extended to refer to this new rewrite relation.

Our goal is to transform any proof $s \leftrightarrow_{E_0 \cup R_0 \cup A}^* t$ into a normalized proof of the form $s \to_{A\backslash R^\infty}^! u \leftrightarrow_A^* v \leftarrow_{A\backslash R^\infty}^! t$, in which s and t are reduced by the above relation to A-equivalent terms u and v. If R^∞ is finite, and an A-matching algorithm is available, then this gives us a means of testing validity in $E_0 \cup R_0 \cup A$. In the set $\mathbf{cp}_A(R)$, we

include all critical pairs obtained by overlapping A-variants of rules in R on rules in R (there is no need to overlap a variant on a variant). That is, $d\mu \leftrightarrow g\mu[r\mu]_p$ is in $\mathbf{cp}_A(R)$ if $g \to d$ and $l \to r$ are rules in R, p is the position of a nonvariable subterm of g, and μ is a most general substitution (most general, with respect to subsumption modulo A) such that $g\mu/p \leftrightarrow_A^* l\mu$, where variables in the two rules have been made disjoint. A non-normal proof might also have a "cliff" of the form $s \leftarrow_R u \leftrightarrow_A t$. By $\mathbf{ex}_A(R)$ we denote the set of extended rules obtained by overlapping variants of rules in R on equations in A. That is, $d\mu \to g\mu[r\mu]_p$ is in $\mathbf{ex}_A(R)$ if $g \leftrightarrow d$ is an equation in A, $l \to r$ is a rule in R, p is the position of a nonvariable subterm of g, and μ is a most general substitution such that $g\mu/p \leftrightarrow_A^* l\mu$, where variables in the two equations have been made disjoint.

Definition 3. A completion sequence in KB/A is *fair* if all persistent critical pairs are considered ($\mathbf{cp}_A(R^\infty)$ is a subset of the A-variants of $\cup E_i$), all persistent critical extensions are considered ($\mathbf{ex}_A(R^\infty)$ is a subset of the A-variants of $\cup R_i$), no simplifiable rule persists (R^∞ is reduced), and no equation persists (E^∞ is empty).

Extended Critical Pair Lemma. *For any rewrite system R, equational system A, and peak or cliff $s \leftarrow_{R\cup A} u \to_{A\setminus R} t$, there exists a rewrite proof $s \to_{A\setminus R}^* v \leftrightarrow_A^* w \leftarrow_{A\setminus R}^* t$, a critical-pair proof $s \leftrightarrow_{A\setminus \mathbf{cp}_A(R)} t$, or extended-rule proof $s \to_{A/\mathbf{ex}_A(R)} v \leftrightarrow_A^* t$ (Jouannaud, 1983).*

The critical-pair and extended-rule proofs may involve A steps below their variable positions only.

Extended Proof Normalization Theorem. *If a completion sequence $(E_0; R_0) \vdash_{KB/A} (E_1; R_1) \vdash_{KB/A} \ldots$ is fair, then for any proof $s \leftrightarrow_{R_i\cup E_i\cup A}^* t$, there is a rewrite proof $s \to_{A\setminus R^\infty}^! u \leftrightarrow_A^* v \leftarrow_{A\setminus R^\infty}^! t$ for R^∞ (Jouannaud and Kirchner, 1986).*

Jouannaud and Kirchner prove this for a specific completion procedure; Bachmair and Dershowitz (1987a) prove it for any implementation of KB/A.

In practice, for the purposes of AC-completion, rules are *flattened* by removing nested occurrences of associative-commutative symbols.

An associative-commutative unification algorithm (Livesey and Siekmann, 1976; Stickel, 1981; Fages, 1987) is used, in place of the standard (syntactic) unification algorithm, to generate the critical pairs in $\mathbf{cp}_{AC}(R_i)$, and associative-commutative matching is used to apply rules. For each rule $f(s,t) \rightarrow r$ in R_i headed by an associative-commutative function symbol f, an extended rule $f(s, f(t,z)) \rightarrow f(r,z)$ is added and flattened out to $f(s,t,z) \rightarrow f(r,z)$; extensions of extended rules are redundant. Analogous with standard completion, the result R^∞ of a successful KB/AC sequence is unique up to AC-variants and renamings of variables, for any given reduction ordering (Lankford and Ballantyne, 1983).

Additional aspects of completion modulo equational theories are considered in Huet (1980), Jouannaud and Kirchner (1986), and Bachmair and Dershowitz (1987a). Huet deals with the *left-linear* case, when only one occurrence of each variable appears on a left-hand side. Jouannaud and Kirchner analyze exactly which critical pairs are necessary when some rules are left-linear and others are not; Bachmair and Dershowitz take the inference rule and proof-pattern approach and generalize previous results. In particular, the well-founded specialization ordering modulo A, used for simplifying left-hand sides, may be separated into a strict subterm ordering modulo A and a proper subsumption ordering (not modulo A). An alternative approach to completion modulo a congruence is to "protect" certain variants of extended rules; for a discussion of the relative merits of extension *vs.* protection, see Jouannaud and Kirchner (1986) and Bachmair (1987). Different approaches are looked at in Göbel (1983), Jouannaud et al. (1983), and Pedersen (1985). The existence of unification algorithms for particular equational theories is surveyed in Siekmann (1984); general-purpose unification procedures are discussed in Kirchner (1985). The question of uniqueness of reduced modulo systems is dealt with in Dershowitz et al. (1988).

3.4. Critical Pair Criteria

As described in Section 2.3, fairness of a completion sequence requires that all persistent critical pairs be generated. and promises success of the sequence. It is possible to require that fewer pairs be generated and still guarantee success; such a weaker fairness

requirement is called a *critical pair criterion.* A criterion is *correct* if we can still ensure that every fair completion sequence (in the weakened sense) leads to rewrite-only proofs in R^∞. Various critical pair criteria have been investigated recently in Winkler and Buchberger (1983), Winkler (1985), Küchlin (1986a), Pottier (1987), Bachmair and Dershowitz (1988), and Kapur et al. (1988).

Two main approaches have been proposed:

(a) The *connectedness* criterion ignores any critical pair $s \leftrightarrow t$ derived from an overlap $s = r\mu \leftarrow_{R_i} l\mu \rightarrow_{R_i} l\mu[d\mu]_p = t$ of rules for which there exists an alternative proof $s \leftrightarrow^*_{R_i \cup E_i} t$ such that each term in the latter proof is smaller vis-a-vis $\rightarrow^+$ than $l\mu$. A special case of such a criterion is used implicitly in the marking scheme of Huet (1981). More generally, it is unnecessary to generate any critical pair for which there already existed a proof smaller under the ordering $>_{kb}$ used in the proof of termination of $\Rightarrow_{KB}$.

(b) The *compositeness* criterion ignores critical pairs for which the overlapped term $l\mu$ can be rewritten at a position strictly below the point p of overlap. A special case of this criterion is used in Lankford and Ballantyne (1979).

AC-completion is closely related to the Gaussian-like elimination methods for solving the word problem for polynomial ideals; see Buchberger (1987) for a discussion. The connectedness criterion was motivated by concepts first appearing in that context (Buchberger (1979)). Critical pair criteria for the *AC* case are looked at in Kapur et al. (1988), Winkler (1984), Bachmair and Dershowitz (1987a; 1988), and Küchlin (1986b).

4. Applications

In this section, we describe some of the varied applications of rewrite methods to equational reasoning. Most of our examples involve associative-commutative completion; for them, we flatten associative-commutative terms and ignore symmetric extensions of rules obtained from the axiom of commutativity. We show only selected

equations and rules, not full computations. Though completion is used in each case to achieve the desired goal, an unfailing completion method could be applied instead.

Let $\mathcal{G}$ denote the variable-free terms in $\mathcal{T}$; such terms are called *ground*. For applications, we are often interested in the *ground confluence* of a given system R. A system has this restricted confluence property, if $s \leftarrow_R^* u \rightarrow_R^* t$, for any ground terms s, t, and u in $\mathcal{G}$, implies the existence of a ground term v in $\mathcal{G}$ such that $s \rightarrow_R^* v \leftarrow_R^* t$. A system is "ground convergent" if it is terminating and ground confluent. A critical pair criterion for the ground case is described in Küchlin (1987).

4.1. Congruence Closures

As was pointed out in Section 3.2, there is, in general, no reduction ordering that can orient all critical pairs. However, when all equations are ground, such an ordering is possible.[10] With such an ordering, failure is impossible. Moreover, completion is guaranteed to succeed after a finite number of steps, resulting in a decision procedure for the initial set of ground equations (Lankford, 1975). Indeed, validity in an equational theory presentable as a finite number of ground equations is decidable (Ackermann, 1954).

Theorem 1. *Given a total reduction ordering on ground terms and a finite set E_0 of ground terms, the result R^∞ of any sequence from $(E_0; \emptyset)$ generated by a fair completion procedure is finite and canonical (Lankford, 1975).*

In other words, ground completion is complete.

Proof. On account of the totality of the ordering, all critical pairs can be oriented, and no sequence can fail. Since there are no variables, the inference rule, **deduce**, is never needed: all critical pairs are covered by **collapse**. If a rule $g \rightarrow d$ overlaps a rule $l \rightarrow r$, the latter collapses into the pair $l[d]_p \leftrightarrow r$, which is oriented into either $l[d]_p \rightarrow r$ or $r \rightarrow l[d]_p$. In either case, the left-hand side of the new rule is smaller under the reduction ordering than the left-hand side

[10] A total "simplification ordering" is needed; see Dershowitz (1982a).

l of the rule it displaces. Since the ordering is well-founded and E_0 is finite, the completion process must terminate. □

For example, the equations

$$\begin{aligned} f(f(f(f(f(c))))) &\leftrightarrow c \\ f(f(f(c))) &\leftrightarrow c \end{aligned}$$

generate the following completion sequence:

$$\begin{aligned} f(f(f(f(f(c))))) &\rightarrow c \\ f(f(f(c))) &\rightarrow c \\ f(f(c)) &\leftrightarrow c \\ f(f(c)) &\rightarrow c \\ f(c) &\leftrightarrow c \\ f(c) &\rightarrow c \end{aligned}$$

The result is the one rule system

$$f(c) \rightarrow c$$

under which any two congruent terms reduce to the same term.

This computation, by completion, of the "congruence closure" of a finite set of ground terms is reminiscent of the algorithm used in Nelson and Oppen (1980). Normal forms serve as names for classes that are eventually "merged" into congruence classes, each named by the minimal term in its class. Adding nonground rules can, of course, cause completion not to succeed within any finite number of steps, as can happen when "demons" are incorporated in the congruence closure computation (cf. Marcus (1984)).

4.2. Meta-Unification

Let E be an equational theory over a set of terms $\mathcal{T}$. An equation $s = t$ is *satisfiable* in the *free model* $\mathcal{T}$ of E, if there is a substitution σ of terms in $\mathcal{T}$ for the variables of s and t such that the equation becomes valid in E, i.e., $s\sigma =_E t\sigma$. In that case we say that s and t are "unifiable" in E via σ, or that $s = t$ has a *solution* σ in E.

Proposition 2. *An equation is satisfiable in the free model of an equational theory E if, and only if, it is satisfiable in all models of E (with nonempty universe).*

Proof. If an equation $s = t$ is satisfiable in the free model of E, then there is a substitution σ of terms for the variables x_i of s and t such that $s\sigma = t\sigma$ is true in all models A of E, i.e., for all assignments ν of values (in the universe of A) to the variables of $s\sigma$ and $t\sigma$, the value $[\![s\sigma]\!]_\nu$ of $s\sigma$ is equal to the value $[\![t\sigma]\!]_\nu$ of $t\sigma$. It follows that for all models A, there is an assignment μ (assigning to each x_i the value $[\![x_i]\!]_\nu$) such that $[\![s]\!]_\mu = [\![t]\!]_\mu$ in A. In other words, the equation $s = t$ is satisfiable in A. □

The completion procedure may be used to solve equations. We have the following theorem showing how to use completion to determine satisfiability:

Theorem 2. *Let R^∞ be the result of a fair completion sequence from $(E; \{eq(x,x) \to T, eq(s,t) \to F\})$, where eq is a new binary symbol, T and F are new constants, and any equality term $eq(u,v)$ is greater (under the given reduction ordering) than F which is greater than T. The equation $s = t$ is solvable in E if, and only if, the contradiction $F \to T$ is a member of R^∞ (Lankford, 1975).*

Proof. If $s\sigma \leftrightarrow_E^* t\sigma$, then (by fairness) the proof $T \leftarrow eq(s\sigma, s\sigma) \leftrightarrow_E^* eq(s\sigma, t\sigma) \to F$ will eventually simplify to a rewrite proof $T \to_{R^\infty}^! v \leftarrow_{R^\infty}^! F$. Since T is minimal, and F cannot be larger than any (equality) term it is paired with, there must be a rule $F \to T$ in R^∞. □

Since the equality symbol does not appear in E, the left-hand side of rules generated from $eq(s,t) \to F$ can never overlap a subterm of a left-hand side of a rule descending solely from E. Note that the fairness criteria need only ensure that R^∞ is ground confluent, since a proof of contradiction from $E \cup \{eq(x,x) \to T, eq(s,t) \to F\}$ containing only ground terms is always possible when the equation is satisfiable (and $\mathcal{G}$ is nonempty).

For example, say one wishes to find x and y such that $x \cdot y = 1$ in commutative group theory, where $\cdot$ is associative and commutative. Completion can proceed as follows:

$$
\begin{array}{rcl}
1 \cdot x & \leftrightarrow & x \\
x \cdot x^{-} & \leftrightarrow & 1 \\
eq(x,x) & \rightarrow & T \\
eq(x \cdot y, 1) & \rightarrow & F(x,y)
\end{array}
$$

$$
\begin{array}{rcl}
1 \cdot x & \rightarrow & x \\
eq(y,1) & \leftrightarrow & F(1,y) \\
eq(y,1) & \rightarrow & F(1,y) \\
F(1,1) & \rightarrow & T \\
x \cdot x^{-} & \rightarrow & 1 \\
eq(1,1) & \rightarrow & F(x,x^{-}) \\
F(x,x^{-}) & \rightarrow & T \\
eq(1,1) & \rightarrow & F(y^{-},y) \\
F(y^{-},y) & \rightarrow & T \\
1^{-} & \rightarrow & 1 \\
x^{--} & \rightarrow & x \\
x \cdot x^{-} \cdot y & \rightarrow & y \\
(x \cdot y)^{-} & \rightarrow & x^{-} \cdot y^{-}
\end{array}
$$

Variables have been placed as arguments to F to keep track of the satisfying substitutions. Completion thus generates three solutions: $\sigma_1 = \{x \mapsto 1, y \mapsto 1\}$, $\sigma_2 = \{y \mapsto x^{-}\}$, and $\sigma_3 = \{x \mapsto y^{-}\}$. In the process, completion has also generated the following canonical system $AC\backslash G$ for commutative groups:

$$
\begin{array}{rcl}
1 \cdot x & \rightarrow & x \\
x \cdot x^{-} & \rightarrow & 1 \\
1^{-} & \rightarrow & 1 \\
x^{--} & \rightarrow & x \\
x \cdot x^{-} \cdot y & \rightarrow & y \\
(x \cdot y)^{-} & \rightarrow & x^{-} \cdot y^{-}
\end{array}
$$

For each of the solutions σ_i, we have $(x \cdot y)\sigma_i \rightarrow_{AC\backslash G} 1$.

As pointed out in Fages (1983a), insisting on simplifying rules may result in equations or rules that determine recurrence relations

between solutions, rather than an explicit, complete set of solutions. Consider, for example, the following completion sequence

$$\begin{array}{rcl} g(f(y)) & \rightarrow & g(y) \\ eq(g(z), g(0)) & \rightarrow & F(z) \\ eq(x, x) & \rightarrow & T \\ \hline F(0) & \rightarrow & T \\ eq(g(y), g(0)) & \leftrightarrow & F(f(y)) \\ F(f(y)) & \rightarrow & F(y) \end{array}$$

The two generated rules implicitly determine the (complete) set of solutions for z in $g(y) = g(0)$: $\{z \mapsto 0, z \mapsto f(0), z \mapsto f(f(0)), \ldots\}$. This aspect of completion is exploited in Réty et al. (1985).

If instead of equations E, completion starts off with a convergent set of rules R and a soluble goal $eq(s,t) \rightarrow F(x_1, \ldots, x_n)$, where the x_i are the variables in s and t, then it cannot fail before finding a solution and deriving a contradiction. All critical pairs needed for a proof of contradiction are of the form $eq(u,v) \leftrightarrow F(w_1, \ldots, w_n)$ and are orientable from left to right. (Since R is confluent, there is no need to generate critical pairs between its rules.) The effect of completion, in this case, is to *narrow* s and t until a substitution σ is generated which equates the two terms in the theory presented by R. Thus, narrowing, as defined by Slagle (1974) and used for this purpose by Fay (1979) and Lankford and Ballantyne (1979), is a restricted form of completion (as was understood in Lankford (1975) and is noted in Dershowitz (1985a) and Réty et al. (1985)). Since the equality symbol is presumed not to appear in R, any critical pair between a rule $l \rightarrow r$ in R and a generated rule $eq(u,v) \rightarrow F(w_1, \ldots, w_n)$, must involve an overlap of l on a (not necessarily proper) nonvariable subterm u/p of u or v/p of v. The resultant critical pair can be oriented into a new rule, $eq(u[r\mu]_p, v\mu) \rightarrow F(w_1\mu, \ldots, w_n\mu)$ or $(u\mu, v[r\mu]_p) \rightarrow F(w_1\mu, \ldots, w_n\mu)$. This step corresponds exactly to narrowing, as defined in Hullot (1980b); using R to collapse the left-hand side (which is optional from the point of view of generating contradictions) gives the result of Fay's narrowing operation. As indicated by Dershowitz (1985a) and others, ground confluence of R suffices for the purposes of narrowing.

4.3. Synthesis

Like other theorem-proving methods, completion and its variants can be applied to the task of program synthesis from specifications (Dershowitz, 1985a,b). The completion procedure itself *folds* definitions, introducing recursive calls, thereby. Thus, program synthesis may be thought of as generating an executable (usually recursive) definition of the set of solutions to a given specification. (Cf. the methodologies of, e.g., Burstall and Darlington (1977), Manna and Waldinger (1980), Clark (1981), and Hogger (1981).)

Assume that we wish to synthesize a rewrite system for some function $f(x_1, \ldots, x_n)$. The synthesized system must reduce any term composed of the *defined* function f and primitive symbols to a term consisting wholly of primitives. The following definition is useful:

Definition 4. A term t is *ground reducible* for a rewrite system R over a set of terms $\mathcal{T}$ if every ground instance of t in $\mathcal{T}$ is rewritable by R.

A *program* for f, then, is a rewrite system each rule of which follows from the specification of f and for which the term $f(x_1, \ldots, x_n)$ is ground reducible (the x_i are variables).

In another context, it has been shown that

Proposition 3. *Ground reducibility is decidable when R is finite (Plaisted, 1985b).*

See also Kapur et al. (1987). When R is left-linear, then deciding ground reducibility can be done more efficiently (Jouannaud and Kounalis, 1986). In the left-linear case, ground reducibility is known to be decidable for rewriting modulo AC as well. Kounalis (1985) considers rewriting modulo a congruence, in general. In the special case where all irreducible forms are constructor terms, ground reducibility is more easily testable. This case had been considered by Nipkow and Weikum (1982) (for left-linear systems) and Dershowitz (1985a) (for the general case), using the idea of a "test set," first suggested in Plaisted (1980). Sufficient criteria had been given earlier in Bidoit (1981), Huet and Hullot (1982), and Padawitz (1983); see also Thiel (1984). New results are in Comon (1986) and Kapur et al. (1986). Ground reducibility has important implications for the question of "sufficient completeness" of algebraic specifications

of data types (as defined in Guttag and Horning (1978)); see the above references.

The completion procedure is given a set E of equations, specifying the required properties of f and necessary properties of the problem domain. The reduction ordering supplied to the procedure should ensure that terms containing specification symbols are greater than corresponding terms containing the defined symbol, which in turn should be greater than the corresponding primitive terms. This serves to guarantee that specification symbols will not appear in the synthesized program.

Theorem 3. *Suppose there exists a rewrite program for a defined function f that is valid for some specification E of f and is contained in a reduction ordering $\succ$. Then the result R^∞ of any fair completion sequence from $(E;\emptyset)$, using $\succ$, will contain a rewrite program for f (Dershowitz, 1985a).*

Proof. Let R constitute such a program for f. Both sides of all identities of E reduce under R^∞ to the same term. In particular, at some stage n, all rules $l \to r$ of R will follow from R_n. At that point, $l \to^!_{R_n} v \leftarrow^!_{R_n} r$ and any term rewritable by R is also rewritable by R_n, since the reduction ordering precludes $r \to^+_{R_n} l$. In particular, all ground terms containing f are reducible. Thus, R_n contains a program for f. □

For example, consider the specification

$$x + x \quad \to \quad double(x)$$

of a function to double a nonnegative integer and suppose we are given the following facts about addition:

$$\begin{aligned} x + 0 &\to x \\ x + s(y) &\to s(x + y) \end{aligned}$$

where + is associative and commutative. The defined symbol symbol is *double*; it is specified using +. The generated system must make *double* (x) ground reducible: it must compute normal forms of ground terms (not containing +) using only the primitive function s and constant 0. The completion sequence

$$\begin{aligned} double(0) &\leftrightarrow 0 \\ double(0) &\rightarrow 0 \\ double(s(y)) &\leftrightarrow s(s(y+y)) \\ double(s(y)) &\leftrightarrow s(s(double(y))) \\ double(s(y)) &\rightarrow s(s(double(y))) \end{aligned}$$

includes the two rule program

$$\begin{aligned} double(0) &\rightarrow 0 \\ double(s(y)) &\rightarrow s(s(double(y))) \end{aligned}$$

Any ground term constructed from *double*, s, and 0 must contain an instance of one of these left-hand sides.

Were + not asserted to be commutative, completion would slowly generate an infinite number of specific values, e.g. $double\,(s(s(0))) \rightarrow s(s(s(s(0))))$, despite the existence of the above, finite program. In general, rules in a program need not be true in all models of the specifications, only in the "intended" model. Thus, verifying the correctness of a rewrite system (such as this one for noncommutative +) really amounts to proving the consistency of the program with its specifications. Completion-based methods for consistency proofs are described below; synthesizing programs whose rules are not deductive, equational consequences, but are, rather, inductive consequences, is investigated in Dershowitz and Pinchover (1989).

A (ground) convergent system may also be used to narrow equational goals, in the spirit of "logic programming" (Kowalski, 1974). Narrowing-based programming languages have been proposed by Dershowitz and Josephson (1984), Goguen and Meseguer (1984), Dershowitz (1985a), Fribourg (1985), Reddy (1985), and others. In Dershowitz (1985b), the synthesis of logic programs using the completion procedure is also illustrated.

4.4. Proof by Consistency

Musser (1980) first suggested using completion to prove theorems in the initial model of an equational theory; such proofs normally require structural induction (see Goguen (1980)). This approach

has been dubbed "inductionless induction" by Lankford (1981) and "proof by consistency" by Kapur and Musser (1987).

An equation $s = t$ is valid in the *initial algebra* $I(E)$ (the "standard model") of an equational theory E if it holds, in E, for all substitutions σ of ground terms for its variables. We write $s =_{I(E)} t$ to indicate that $s\sigma =_E t\sigma$ for all ground σ. Goguen (1980) showed that for any convergent system R, the ground normal forms constitute an algebra that is initial for the theory presented by R. In that sense rewriting is a "correct" implementation of initial algebra semantics.

Were a convergent system R to include rules that reduce any valid ground equation $eq(g, d)$ to T and invalid ones to F (for some equality symbol eq), then letting $(E_0; R_0)$ be $(\{s \leftrightarrow t\}; R)$ and completing fairly, would generate the contradiction $F \rightarrow T$ whenever $s = t$ is not an identity in the initial model of R. This is the method of Musser (1980) (see also Goguen (1980), Huet and Oppen (1980), and Kapur (1980)); its correctness follows directly from the fairness of completion, since, if $s \neq_{I(R)} t$, then $F \leftarrow^!_{R_0} eq(s\sigma, t\sigma) \leftrightarrow^*_{E_0} eq(t\sigma, t\sigma) \rightarrow^!_{R_0} T$, for some ground σ, would be a proof of a contradiction. Huet and Hullot (1982) refined this method for the important case when the set of ground normal forms is the set of all ground terms constructed from a restricted set $\mathcal{C} \subset \mathcal{F}$ of function symbols, called *constructors*, i.e., for which the constructor algebra is initial. A contradiction, then, is indicated by any clash between constructors. The inference rule

Contradiction:

$$(E \cup \{f(s_1, \ldots, s_n) \leftrightarrow g(t_1, \ldots, t_n)\}; R) \vdash (\emptyset; \{F \rightarrow T\}) \quad \text{if } f! = g \text{ and } f, g \in \mathcal{C}$$

may be added (explicitly or implicitly) to completion. In addition, the inference rule

Decompose:

$$(E \cup \{f(s_1, \ldots, s_n) \leftrightarrow f(t_1, \ldots, t_n)\}; R) \vdash (E \cup \{s_i \leftrightarrow t_i : i = 1, \ldots, n\}; R) \quad \text{if } f \in \mathcal{C}$$

can be used to speed up the proof process. This idea was generalized in Paul (1984).

Lankford (1981) formalized the method: one proves consistency by showing that the ground normal forms of a convergent system R for E and the result R^∞ of completion are the same. (Lankford includes restrictions on the form of rules, inherited from Huet and

Hullot.) Dershowitz (1982b) and Rémy (1982) first noted the connection between inductive theorem proving and the notion of reducibility of all ground instances.[11] Jouannaud and Kounalis (1986) recognized the centrality of this notion and coined the term "inductive reducibility";[12] we prefer to use "ground re ducibility" (suggested by Küchlin (1987) and already used above) for this concept.

Proposition 4. *Let an equational theory E admit a convergent rewrite system R. An equation $s = t$ is valid in the initial model $I(E)$ of E if, and only if, no equality between two distinct ground R-normal forms follows from E and $s \leftrightarrow t$ (Dershowitz (1982b)).*

This result extended the original method by allowing any contradictory equality between irreducible ground terms to indicate invalidity.

Proof. If $s\sigma \neq_E t\sigma$, for some ground substitution σ, then the R-normal forms of $s\sigma$ and $t\sigma$ are distinct ground terms whose equality follows from E and $s \leftrightarrow t$. Conversely, if g and d are ground terms that are equal only if $s = t$, then some ground instance $s\sigma \leftrightarrow t\sigma$ of $s \leftrightarrow t$, that does not hold in E alone, must be necessary for a proof of the inconsistency $g = d$. □

Theorem 4. *Let H be a set of equational hypotheses. Suppose R is ground convergent for E and let R^∞ be the result of a fair completion sequence starting from $(H; R)$. Every left-hand side of R^∞ is ground R-reducible if, and only if, $s =_{I(E)} t$ for all $s \leftrightarrow t$ in H (Dershowitz (1982b)).*

This was first proved for convergent R; Rémy and Zhang (1984) pointed out that only ground confluence is needed.

Proof. Suppose $l\sigma$ is a ground instance of a left-hand of a rule $l \to r$ in R^∞ that is R-irreducible. Since $R \cup R^\infty$ is terminating, the R-normal form of $r\sigma$ cannot be $l\sigma$ and an inconsistency exists. On the other hand, if $s \to^!_{R_\infty} v \leftarrow^!_{R_\infty} t$ for two distinct R-irreducible ground terms s and t, then at least one of them

[11] Rémy's (1982) conditions for inconsistency confused reducibility (in general) with ground reducibility; see, however, Kirchner (1984).

[12] In earlier work, they called it "quasi-reducibility."

must be reducible by a rule in R^∞ with a ground reducible left-hand side. □

If the completion procedure yields a (possibly infinite) convergent rewrite system R^∞ such that the previously irreducible ground terms are still irreducible, then the equations in H are valid in the initial model. Of course, when R^∞ is infinite, testing for ground reducibility is no longer feasible. If completion ever generates an equation (not just a rule) that has as an instance an equality between two distinct irreducible ground terms, then too H is not valid in $I(E)$; if the procedure fails without producing such an equation, then nothing can be said about H (Dershowitz (1982b)). An extension of this method to proving non-orientable equations is described in Jouannaud and Kounalis (1986).

This method applies to AC-completion as well (see Goguen (1980) and Lankford (1981)), providing the basis for the experiments reported in Huet and Oppen (1980) and Huet and Hullot (1982). More in this area is included in Jouannaud and Kounalis (1986). Kirchner (1984) considers the general completion modulo A case. The proof by consistency method has also been generalized in Kapur and Musser (1986, 1987); see also Paul (1984). In Plaisted (1982) it was shown how completion may be used to prove the existence of initial algebra semantics for error conditions.

As an example, consider the canonical system $AC\backslash R$ containing:[13]

$$
\begin{array}{rcl}
x + 0 & \rightarrow & x \\
x \times 0 & \rightarrow & 0 \\
x \times 1 & \rightarrow & x \\
x \times (y + z) & \rightarrow & (x \times y) + (x \times z) \\
\sum_{i=1}^{0} i & \rightarrow & 0 \\
\sum_{i=1}^{1} i & \rightarrow & 1 \\
\sum_{i=1}^{x+1} i & \rightarrow & \sum_{i=1}^{x} i + (x + 1)
\end{array}
$$

[13] From Huet and Oppen (1980).

where both + and × are associative and commutative. Adding the hypothesis

$$(1+1) \times \sum_{i=1}^{x} i \quad \leftrightarrow \quad x \times (x+1)$$

and completing, generates the rules

$$(\sum_{i=1}^{x} i) + (\sum_{i=1}^{x} i) \quad \rightarrow \quad (x \times x) + x$$

$$(\sum_{i=1}^{x} i) + (\sum_{i=1}^{x} i) + z \quad \rightarrow \quad (x \times x) + x + z$$

(the second is an extension of the first), but no more. Since the left-hand sides are both ground reducible by $\rightarrow_{AC \setminus R}$, the theorem is proved.

Proof by consistency has been proposed as an alternative to other automated approaches to inductive theorem proving. Indeed, in the opinion of this writer (see also the discussion in Fribourg (1987)), it bears a measure of resemblance with some of the heuristics incorporated in the methodology of Boyer and Moore (1979). "Cross-fertilization" of goals—in the latter's terminology—corresponds roughly with critical pair generation; "unfolding," with equation simplification; and "throwing away equalities," with rule simplification. The reduction ordering supplied to completion suggests the "induction scheme" and restricts the locations for induction to variable occurrences on the left-hand side. But completion has some drawbacks, notably its insistence on simultaneously "inducting" on all "unflawed" occurrences of variables on left-hand sides. Fribourg (1986) points out that not all critical pairs need be considered (since ground confluence depends only on ground terms and since R itself is confluent) and that fairness need only guarantee ground confluence of R^{∞}; see also Göbel (1985). In particular, only critical pairs that involve a selected induction term in an hypothesis need be generated (provided that critical pairs at that position will cover all ground instances). Göbel (1987) and Küchlin (1987) adapt the subconnectedness criterion to the ground case. These results may allow for better-tailored proofs by consistency. An unfailing version and improved results in this area are contained in Bachmair (1988).

Suppose we are given the system

$x + 0$	$\rightarrow$	x
$x + s(y)$	$\rightarrow$	$s(x+y)$

and try to prove associativity:

$$(x + y) + z \quad \leftrightarrow \quad x + (y + z).$$

(Commutativity of + is also unknown.) Depending on the reduction ordering, completion may or may not prove the hypothesis. On the one hand, we have the successful sequence:

$$\begin{aligned}
x + (y + z) &\rightarrow (x + y) + z \\
x + y &\leftrightarrow (x + y) + 0 \\
x + y &\leftrightarrow x + y \\
x + s(y + z) &\leftrightarrow (x + y) + s(z) \\
x + s(y + z) &\leftrightarrow s((x + y) + z) \\
s(x + (y + z)) &\leftrightarrow s((x + y) + z) \\
s((x + y) + z) &\leftrightarrow s((x + y) + z)
\end{aligned}$$

where all generated equations simplify to a trivial one. On the other hand, with a different ordering, we have an infinite sequence:

$$\begin{aligned}
(x + y) + z &\rightarrow x + (y + z) \\
s(x + y) + z &\leftrightarrow x + (s(y) + z) \\
s(x + y) + z &\rightarrow x + (s(y) + z) \\
s(s(x + y)) + z &\rightarrow x + (s(s(y)) + z) \\
&\vdots
\end{aligned}$$

One might argue that the first choice of ordering is more appropriate, since—in orienting rules—it only considers the second argument of +, that argument on which the recursive definition of + is based. Alternatively, once can place the blame on the overlapping of the second rule for + on the y variable in the hypothesis. The method of Fribourg (1986) could, therefore, be used to only generate critical pairs at the position of x, as was the case with the first ordering.

As pointed out in Lankford (1981), neither completion nor other inductive methods are good at coming up with the theorem itself, or suggesting lemmata needed for its proof. Some hope for progress in this direction may lie in the recent work of Kirchner (1987) on inferring meta-rules from infinite completion sequences. See also Dershowitz and Pinchover (1989).

4.5. First-Order Theorem-Proving

Hsiang (1982) first suggested using a canonical rewrite system for Boolean algebra in a resolution-like theorem-proving strategy. Let $AC\backslash BA$ be the following canonical system:[14]

$\neg x$	$\rightarrow$	$x \oplus T$
$x \vee y$	$\rightarrow$	$(x \wedge y) \oplus x \oplus y$
$x \supset y$	$\rightarrow$	$(x \wedge y) \oplus x \oplus T$
$x \wedge T$	$\rightarrow$	x
$x \wedge F$	$\rightarrow$	F
$x \wedge x$	$\rightarrow$	x
$x \oplus F$	$\rightarrow$	x
$x \oplus x$	$\rightarrow$	F
$(x \oplus y) \wedge z$	$\rightarrow$	$(x \wedge z) \oplus (y \wedge z)$
$x \wedge x \wedge y$	$\rightarrow$	$x \wedge y$
$x \oplus x \oplus y$	$\rightarrow$	y

where $\neg$ is "not," $\wedge$ is "and," $\vee$ is "inclusive-or," $\oplus$ is "exclusive-or," $\supset$ is "implication," T is "true," and F is "false." Both $\wedge$ and $\oplus$ are associative and commutative. (The last two are extended rules.) With this system, all propositional tautologies reduce to T and contradictions, to F. The following, for example, is a rewrite proof of DeMorgan's Law:

$$
\begin{aligned}
&(\neg x) \vee (\neg y) \\
&\quad \rightarrow^*_{BA} (x \oplus T) \vee (y \oplus T) \\
&\quad\quad \rightarrow_{BA} ((x \oplus T) \wedge (y \oplus T)) \oplus x \oplus T \oplus y \oplus T \\
&\quad\quad\quad \rightarrow_{BA} (x \wedge (y \oplus T)) \oplus (T \wedge y) \oplus T)) \oplus x \oplus T \oplus y \oplus T \\
&\quad\quad\quad\quad \rightarrow^*_{AC\backslash BA} (x \wedge y) \oplus (x \wedge T) \oplus (T \wedge y)
\end{aligned}
$$

[14] This is based on Boolean rings as in Stone (1936); cf. Watts and Cohen (1980). The exclusive-or normal form has also been credited to Zhegalkin (1927), who also worked on Boolean rings. The two equational theories, viz. Boolean rings with unit and Boolean algebras, are equivalent, in that the operations of one can be defined in terms of those of the other.

$$\begin{aligned}&\oplus (T \wedge T) \oplus x \oplus T \oplus y \oplus T\\ &\quad \rightarrow^*_{AC\backslash BA} (x \wedge y) \oplus x \oplus y \oplus T \oplus x \oplus T \oplus y \oplus T\\ &\quad \rightarrow_{AC\backslash BA} (x \wedge y) \oplus T \leftarrow_{BA} \neg(x \wedge y)\end{aligned}$$

Note that by incorporating the logical capabilities of $AC\backslash BA$ with the techniques of meta-unification, Boolean combinations of equations may be solved.

A formula r in first-order predicate calculus is valid, if, and only if, the closed Skolemized version $(\forall v_1 \ldots \forall v_n)s$ of its negation $\neg r$ is false under Herbrand interpretations.[15]

Theorem 5. *Let R^∞ be the result of a fair AC-completion sequence starting from $(\{s \leftrightarrow T\}; BA)$, where s is a quantifier-free formula and any (non-trivial) formula is greater (under the given reduction ordering) than F which is greater than T. Then the closed formula $(\forall v_1 \ldots \forall v_n)s$ is unsatisfiable if, and only if, R^∞ contains the contradiction $F \rightarrow T$ (Hsiang and Dershowitz (1983)).*

Proof. If the formula is unsatisfiable, then, by Herbrand's Theorem, a finite conjunction of instances $s\sigma_i$ of s reduces to F under $AC\backslash BA$. Hence, we have

$$F \leftarrow^*_{AC\backslash BA} s\sigma_1 \wedge \cdots \wedge s\sigma_n \leftrightarrow^* T \wedge \cdots \wedge T \rightarrow^*_{BA} T$$

and, by fairness, completion will generate rules under which F reduces to T. The only possible such rule is $F \rightarrow T$. □

For example, to prove that

$$(\forall y \exists z)\neg[p(z, y) \equiv (\forall x)\neg(p(z, x) \wedge p(x, z))]$$

is valid, one can show that its Skolemized negation,

$$\begin{aligned}(\forall z \forall x)(&\{p(z, c) \supset \neg[p(z, x) \wedge p(x, z)]\}\\ &\wedge \{[p(z, f(z)) \wedge p(f(z), z)] \vee p(z, c)\}),\end{aligned}$$

[15] *Skolemization* is the process of replacing an existentially quantified subformula of the form $(\exists v)t$ with $t[f(u_1, \ldots, u_m)]$, where the u_i are the universally quantified variables in t whose scope includes $(\exists v)t[v]$ and *all* occurrences of v in t bound by the deleted quantifier $(\exists v)$ have been replaced by $f(u_1, \ldots, u_m)$. The formula is then *closed* by adding universal quantifiers $(\forall v)$ for all free variables v. (See, e.g., Chang and Lee, 1973.)

is unsatisfiable, where f is a unary Skolem function and c, a Skolem constant. Adding the above assertion to the BA system and completing gives[16]

$$
\begin{aligned}
&\{p(z,c) \supset [p(z,x) \wedge p(x,z)]\} \\
&\quad \wedge \{[p(z,f(z)) \wedge p(f(z),z)] \vee p(z,c)\} && \leftrightarrow && T \\
&[p(z,c) \wedge p(z,x) \wedge p(x,z)] \\
&\quad \oplus [p(z,f(z)) \wedge p(f(z),z)] \wedge p(z,c) \\
&\quad \oplus [p(z,f(z)) \wedge p(f(z),z)] \oplus p(z,c) && \rightarrow && T \\
&\quad [p(f(z),z) \wedge p(z,f(z))] \oplus p(z,c) && \rightarrow && T \\
&[p(f(z),z) \wedge p(z,f(z))] \oplus p(z,c) \oplus y && \rightarrow && T \oplus y \\
&p(f(z),z) \wedge p(z,f(z)) && \rightarrow && T \oplus p(z,c) \\
&[p(z,c) \wedge p(z,x) \wedge p(x,z)] \oplus T && \rightarrow && T \\
&[p(y,c) \wedge p(c,y)] \oplus T && \rightarrow && T \\
&T \oplus p(c,c) \oplus T && \rightarrow && T \\
&p(c,c) && \rightarrow && T \\
&[T \wedge p(c,c)] \oplus T && \rightarrow && T \\
&p(c,c) \oplus T && \rightarrow && T \\
&T \oplus T && \rightarrow && T \\
&F && \rightarrow && T \\
&x && \leftrightarrow && T
\end{aligned}
$$

Notice that the inference rules (**deduce**, **collapse**, and **orient**) automatically replace a rule of the form $s \oplus t \rightarrow u$ with $s \rightarrow t \oplus u$, whenever $s \succ t \oplus u$. Also, a rule of the form $s \wedge t \rightarrow T$ begets $s \rightarrow T$ and $t \rightarrow T$. Thus, with a reasonable ordering, $s \vee t \rightarrow T$ would be replaced by $s \wedge t \rightarrow s \oplus t \oplus T$ and $s \supset t \rightarrow T$ by $s \wedge t \rightarrow s$.

The above method is not refutationally complete, however, because of the potential failure of a completion sequence. For example, with the trivial ordering (T less than anything else) and the inconsistent set of rules

$$
\begin{aligned}
p \oplus q \oplus T &\rightarrow T \\
p \oplus r \oplus T &\rightarrow T \\
q \oplus r &\rightarrow T
\end{aligned}
$$

[16] For a comparable proof by resolution, see Example 2–38 in Manna (1974).

completion will generate unorientable equations like

$$q \oplus T \leftrightarrow r \oplus T,$$

but not the contradiction $F \rightarrow T$. For completeness, an unfailing version would be required. Hsiang (1985) has shown how converting C to its clausal form $C_1 \wedge C_2 \wedge \cdots \wedge C_n$, adding a rule $C_i \oplus T \rightarrow F$ for each clause C_i, and then running the AC-completion procedure with the trivial order is guaranteed not to fail. The proof of completeness, however, does not fully cover simplification of rules. Hsiang also gave a criterion for disregarding critical pairs that do not derive from at least one rule that is a sum, and for restricting the amount of associative-commutative unification necessary in computing critical pairs. Related work is in Paul (1985). An important aspect of Hsiang's approach is the ability to incorporate convergent systems for specific domain theories; see also Fages (1983b). A general extension to first-order predicate calculus with equality is in Hsiang (1987). Kapur and Narendran (1985a) propose a nonclausal method that uses BA and includes inferences based on cancellation; its refutational completeness needs to be established. The additional inference rule is

Cancel: $(E \cup \{s \oplus u \leftrightarrow t \oplus u\}; R) \vdash (E \cup \{s \leftrightarrow t\}; R)$

Bachmair and Dershowitz (1987b) present a method that incorporates simplification and prove it complete.

5. Conclusion

In this chapter, we have examined the completion procedure, some of its multifaceted extensions, and its main areas of application. We have viewed the procedure and its extensions as inference engines which use a limited amount of "forward reasoning" (i.e., deduction), trying to churn out enough equational consequences of the given axioms to guarantee that for any equational proof, there will eventually be one that uses only "backward reasoning" (i.e., reduction). The applications highlight the advantages inherent in the sense of direction gained from the incorporation of a well-defined notion of "simplicity."

Recently, the completion concept has been further extended to "conditional" equations and rules, where the applicability of an identity or rewrite is predicated on the fulfillment of some condition(s).

Conditional equations are very important in applications, such as data type specification and functional programming languages. A pioneering study of conditional rewriting was Brand et al. (1978); other works in this area are Rémy (1982) and Kaplan (1984) (see Kaplan and Rémy (1989)); an overview of results on the Church-Rosser property for conditional systems may be found in Dershowitz et al. (1988). Various completion-like approaches to conditional equations and Horn clauses are described in Brown (1975), Jouannaud and Waldmann (1986), Paul (1986), Ganzinger (1987), Kounalis and Rusinowitch (1987), Bachmair et al. (1989), and Kaplan (1988). More general refutationally-complete combinations of resolution and completion are described in Lankford (1975), Peterson (1983), Hsiang and Rusinowitch (1986), and Rusinowitch (1987).

Acknowledgement

I gratefully acknowledge the comments of my friends, Leo Bachmair, Jieh Hsiang, Jean-Pierre Jouannaud, Dallas Lankford, Pierre Lescanne, David Plaisted, Jean-Luc Rémy, and Mitsuhiro Okada.

References

Ackermann, W. (1954). *Solvable Cases of the Decision Problem.* North-Holland, Amsterdam.

Avenhaus, J. (1985). "On the termination of the Knuth–Bendix completion algorithm," Report 120/84, Universität Kaiserslautern, Kaiserslautern, West Germany.

Bachmair, L. (1987). "Proof methods for equational theories," Ph.D. Thesis, Department of Computer Science, University of Illinois, Urbana, Ill.

Bachmair, L. (1988). "Proof by consistency in equational theories," *Proceedings of the Third Symposium on Logic in Computer Science*, Edinburgh, Scotland, July 1988, 228–233.

Bachmair, L., and Dershowitz, N. (1987a). "Completion for rewriting modulo a congruence," *Proceedings of the Second International Conference on Rewriting Techniques and Applications*, Bordeaux, France, May 1987, 192–203. (Available as Vol. 256, *Lecture Notes in Computer Science*, Springer, Berlin; revised version to appear in *Theoretical Computer Science*, 1988).

Bachmair, L., and Dershowitz, N. (1987b). "Inference rules for rewrite-based first-order theorem proving," *Proceedings of the Second Symposium on Logic in Computer Science*, Ithaca, NY, June 1987, 331–337.

Bachmair, L., and Dershowitz, N. (1988). "Critical pair criteria for completion," *J. Symbolic Computation* **6** (1), 1–18. (Previous version appeared as "Critical

pair criteria for the Knuth–Bendix completion procedure," *Proceedings of the 1986 ACM Symposium on Symbolic and Algebraic Computation*, Waterloo, Ontario, July 1986, 215–217.)

Bachmair, L., Dershowitz, N., and Hsiang, J. (1986). "Orderings for equational proofs," *Proceedings of the Symposium on Logic in Computer Science*, Cambridge, MA, June 1986, 346–357.

Bachmair, L., Dershowitz, N., and Plaisted, D. A. (1989). "Completion without failure," *Resolution of Equations in Algebraic Structures*. Academic Press, Boston.

Benninghofen, B., Kemmerich, S., and Richter, M. M. (1987). "Systems of Reductions," *Lecture Notes in Computer Science* **277**, Springer, Berlin.

Bidoit, M. (1981). "Une méthode de présentation de types abstraits: Applications," Thèse, Université de Paris-Sud, Orsay, France.

Birkhoff, G. (1935). "On the structure of abstract algebras," *Proceedings of the Cambridge Philosophical Society* **31**, 433–454.

Boyer, R. S., and Moore, J. S. (1979). *A Computational Logic*. Academic Press, New York.

Brand, D., Darringer, J. A., and Joyner, W. J., Jr. (1978). "Completeness of conditional reductions," Report RC 7404, IBM T. J. Watson Research Center, Yorktown Heights, NY.

Brown, T. C., Jr. (1975). "A structured design-method for specialized proof procedures," Ph.D. Thesis, California Institute of Technology, Pasadena, CA.

Buchberger, B. (1979). "A criterion for detecting unnecessary reductions in the construction of Gröbner bases," *Proceedings of the European Conference on Symbolic and Algebraic Computing*, 3–21. (Available as Vol. 72, *Lecture Notes in Computer Science*, Springer, Berlin.)

Buchberger, B. (1987). "History and basic features of the critical-pair/completion procedure," *J. Symbolic Computation* **3** (1 & 2), 3–38. (Previous version appeared as "Basic features and development of the critical pair/completion procedure," *Proceedings of the First International Conference on Rewriting Techniques and Applications*, Dijon, France, May 1985, 1–45.)

Burstall, R. M., and Darlington, J. (1977). "A transformation system for developing recursive programs," *J. of the Association for Computing Machinery* **24**(1), 44–67.

Butler, G., and Lankford, D. S. (1980). "Experiments with computer implementations of procedures which often derive decision algorithms for the word problem in abstract algebras," Memo MTP-7, Department of Mathematics, Louisiana Tech. University, Ruston, LA.

Chang, C. L., and Lee, R. C. (1973). *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, New York.

Clark, K. L. (1981). "The synthesis and verification of logic programs," Research Report DOC 81/36, Department of Computing, Imperial College, London, England.

Comon, H. (1986). "Sufficient completeness, term rewriting systems and 'anti-unification'," *Proceedings of the Eight International Conference on Automated Deduction, Oxford, England*. (Available as Vol. 230, *Lecture Notes in Computer Science*, Springer, Berlin.)

Curry, H. B., and Feys, R. (1958). *Combinatory Logic*. North-Holland, Amsterdam.

Davis, M. (1958). *Computability and Unsolvability*. McGraw-Hill, New York.

Dershowitz, N. (1982a). "Orderings for term-rewriting systems," *Theoretical*

Computer Science **17**(3), 279–301. (Previous version in the *Proceedings of the Symposium on Foundations of Computer Science*, San Juan, PR, 123–131, October 1979.)

Dershowitz, N. (1982b). "Applications of the Knuth–Bendix completion procedure," *Proceedings of the Seminaire d'Informatique Theorique*, Paris France, 95–111.

Dershowitz, N. (1985a). "Computing with rewrite systems," *Information and Control* **64**(2/3), 122–157. (Previous version in the *Proceedings of the NSF Workshop on the Rewrite Rule Laboratory*, Schenectady, NY, pp. 269–298, September 1983.)

Dershowitz, N. (1985b). "Synthesis by completion," *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, Los Angeles, CA, 208–214.

Dershowitz, N. (1987). "Termination of rewriting," *J. of Symbolic Computation* **3**(1 & 2), 69–115; **4**(3), 409–410. (Previous version appeared as "Termination," *Proceedings of the First International Conference on Rewriting Techniques and Applications*, Dijon, France, May 1985.)

Dershowitz, N., and Josephson, N. A. (1984). "Logic programming by completion," *Proceedings of the Second International Logic Programming Conference*, Uppsala, Sweden, 313–320.

Dershowitz, N., and Jouannaud, J.-P. (1989). "Rewrite systems," in *Handbook of Theoretical Computer Science* (A. Meyer, M. Nivat, M. Paterson, D. Perrin, eds.). North-Holland, Amsterdam. (To appear.)

Dershowitz, N., and Manna, Z. (1979). "Proving termination with multiset orderings," *Communications of the ACM* **22**(8), 465–476. (Previous version in the *Proceedings of the International Colloquium on Automata, Languages and Programming*, Graz, 188–202, July 1979.)

Dershowitz, N., and Okada, M. (1988). "Proof-theoretic techniques and the theory of rewriting," *Proceedings of the Third Symposium on Logic in Computer Science*, Edinburgh, Scotland, 104–111.

Dershowitz, N., and Pinchover, E. (1989). "Inductive synthesis of equational programs." Submitted.

Dershowitz, N., Marcus, L., and Tarlecki, A. (1988). "Existence, uniqueness, and construction of rewrite systems," *SIAM J. on Computing* **17**(4), 629–639. (Previous version appeared as "Existence and construction of rewrite systems," N. Dershowitz and L. Marcus, The Aerospace Corporation, December 1982.)

Detlefs, D., and Forgaard, R. (1985). "A procedure for automatically proving the termination of a set of rewrite rules," *Proceedings of the First International Conference on Rewriting Techniques and Applications*, Dijon, France, 255–270. (Available as Vol. 202, *Lecture Notes in Computer Science*, Springer, Berlin, September 1985.)

Dick, A. J. J. (1986). "ERIL—Equational reasoning: An interactive laboratory," Report No. RAL86010, Rutherford Appleton Laboratory, Chilton, U.K.

Evans, T. (1951). "On multiplicative systems defined by generators and relations, I," *Proceedings of the Cambridge Philosophical Society* **47**, 637–649.

Fages, F. (1983a). "Note sur l'unification des termes de premier ordre finis et infinis," Rapport 83–29, Laboratoire Informatique Théorique et Programmation, Université de Paris VII, Paris, France.

Fages, F. (1983b). "Formes canoniques dans les algèbres booléennes, et application à la démonstration automatique en logique de premier ordre," Thèse, Université de Paris VI, Paris, France.

Fages, F. (1984a). "Le système KB: manuel de référence: présentation et bibliographie, mise en oeuvre." Report R. G. 10.84, Greco de Programmation, Bordeaux, France.

Fages, F. (1984b). "Associative-commutative unification," *J. Symbolic Computation* **3**(3), 257–275. (Previous version in the *Proceedings of the Seventh International Conference on Automated Deduction*, Napa, CA, 194–208, May 1984.)

Fay, M. (1979). "First-order unification in an equational theory," *Proceedings of the Fourth Workshop on Automated Deduction*, Austin, TX, 161–167.

Fribourg, L. (1985). "SLOG: A logic programming language interpreter based on clausal superposition and rewriting," *Proceedings of the IEEE Symposium on Logic Programming*, Boston, MA, 172–184.

Fribourg, L. (1986). "A strong restriction of the inductive completion procedure," *Proceedings of the Thirteenth EATCS International Conference on Automata, Languages and Programming*, Rennes, France, 105–115. (Available as Vol. 226, *Lecture Notes in Computer Science*, Springer, Berlin.)

Fribourg, L. (1987). "On the use of conditional rewrite rules in classical automated reasoning," *Proceedings of the First International Workshop on Conditional Rewriting*, Orsay, France. (To appear in *Lecture Notes in Computer Science*, Springer, Berlin.)

Ganzinger, H. (1987). *A completion procedure for conditional equational specifications.* Fachbereich Informatik, Universität Dormund, Dortmund, West Germany.

Gnaedig, I. (1987). "Knuth–Bendix procedure and nondeterministic behavior—An example," *Bulletin of the European Association for Theoretical Computer Science* **32**, 86–92.

Göbel, R. (1983). "A completion procedure for globally finite term rewriting systems," *Proceedings of an NSF Workshop on the Rewrite Rule Laboratory*, Schenectady, NY, 155–203. (Available as Report 84GEN008, General Electric Research and Development, April 1984.)

Göbel, R. (1985). "A specialized Knuth–Bendix algorithm for inductive proofs," *Proceedings of the Conference on Combinatorial Algorithms in Algebraic Structures*, Kaiserslautern, West Germany.

Göbel, R. (1987). "Ground confluence," *Proceedings of the Second International Conference on Rewriting Techniques and Applications*, Bordeaux, France, 156–167. (Available as Vol. 256, *Lecture Notes in Computer Science*, Springer, Berlin.)

Goguen, J. A. (1980). "How to prove algebraic inductive hypotheses without induction," *Proceedings of the Fifth Conference on Automated Deduction*, Les Arcs, France, 356–373. (Available as Vol. 87, *Lecture Notes in Computer Science*, Springer, Berlin.)

Goguen, J. A., and Meseguer, J. (1984). "Equality, types, modules and (why not?) generics for logic programming," *Logic Programming* 1 (2), 179–210.

Guttag, J., and Horning, J. J. (1978). "The algebraic specification of abstract data types," *Acta Informatica* **10** (1) 27–52.

Hogger, C. J. (1981). "Derivation of logic programs," *J. of the Association for Computing Machinery* **28**(2), 372–392.

Hsiang, J. (1982). "Topics in automated theorem proving and program generation," Ph.D. Thesis, Report R-82-1113, Department of Computer Science, University of Illinois, Urbana, IL.

Hsiang, J. (1985). "Refutational theorem proving using term-rewriting systems," *Artificial Intelligence* **25**, 255–300.

Hsiang, J. (1987). "Rewrite method for theorem proving in first order theory with equality," *J. of Symbolic Computation* **3**(1 & 2), 133–151. (Previous version appeared as "Two results in term rewriting theorem proving," *Proceedings of the First International Conference on Rewriting Techniques and Applications*, Dijon, France, 301–324, May 1985.)

Hsiang, J., and Dershowitz, N. (1983). "Rewrite methods for clausal and non-clausal theorem proving," *Proceedings of the Tenth EATCS International Conference on Automata, Languages and Programming*, Barcelona, Spain, 331–346. (Available as Vol. 154, *Lecture Notes in Computer Science*, Springer, Berlin.)

Hsiang, J., and Rusinowitch, M. (1986). "A new method for establishing refutational completeness in theorem proving," *Proceedings of the Eighth International Conference on Automated Deduction*, Oxford, England, 141–152. (Available as Vol. 230, *Lecture Notes in Computer Science*, Springer, Berlin.)

Hsiang, J., and Rusinowitch, M. (1987). "On word problems in equational theories," *Proceedings of the Fourteenth EATCS International Conference on Automata, Languages and Programming*, Karlsruhe, West Germany, 54–71. (Available as Vol. 267, *Lecture Notes in Computer Science*, Springer, Berlin.)

Huet, G. (1980). "Confluent reductions: Abstract properties and applications to term rewriting systems," *J. of the Association for Computing Machinery* **27**(4), 797–821. (Previous version in the *Proceedings of the Symposium on Foundations of Computer Science*, Providence, RI, 30–45, October 1977.)

Huet, G. (1981). "A complete proof of correctness of the Knuth–Bendix completion algorithm," *J. of Computer and System Sciences* **23**(1), 11–21.

Huet, G., and Hullot, J.-M. (1982). "Proofs by induction in equational theories with constructors," *J. of Computer and System Sciences* **25**, 239–266. (Previous version in the *Proceedings of the Twenty-First Annual Symposium on Foundations of Computer Science*, Lake Placid, NY, 96–107, October 1980.)

Huet, G., and Oppen, D. C. (1980). "Equations and rewrite rules: A survey," In *Formal Language Theory: Perspectives and Open Problems* (R. Book, ed.). Academic Press, New York, 349–405.

Hullot, J.-M. (1980a). "A catalogue of canonical term rewriting systems," CSL-113, SRI International, Menlo Park, CA.

Hullot, J.-M. (1980b). "Canonical forms and unification," *Proceedings of the Fifth Conference on Automated Deduction*, Les Arcs, France, 318–334.

Jouannaud, J.-P. (1983). "Confluent and coherent sets of reductions with equations: Application to proofs in abstract data types," *Proceedings of the Eighth Colloquium on Trees in Algebra and Programming*, 269–283. (Available as Vol. 59, *Lecture Notes in Computer Science*, Springer, Berlin.)

Jouannaud, J.-P. (1987). "Proof methods for completion procedures," *Presented at the Second International Conference on Rewriting Techniques and Applications*, Bordeaux, France.

Jouannaud, J.-P., and Kirchner, H. (1986). "Completion of a set of rules modulo a set of equations," *SIAM J. on Computing* **15**, 1155–1194. (Previous version in the *Proceedings of the Eleventh ACM Symposium on Principles of Programming Languages*, Salt Lake City, UT, 83–92, January 1984.)

Jouannaud, J.-P., and Kounalis, E. (1986). "Automatic proofs by induction in equational theories without constructors," *Proceedings of the Symposium on*

Logic in Computer Science, Cambridge, MA, 358–366. (Revised version to appear in *Information and Computation.*)

Jouannaud, J.-P., and Waldmann, B. (1986). "Reductive conditional term rewriting systems," *Proceedings of the Third IFIP Working Conference on Formal Description of Programming Concepts*, Ebberup, Denmark.

Jouannaud, J.-P., Kirchner, H., and Rémy, J. L. (1983). "Church–Rosser properties of weakly terminating term rewriting systems," *Proceedings of the Eight International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, 909–915.

Kaplan, S. (1984). "Conditional rewrite rules," *Theoretical Computer Science* **33**, 175–193.

Kaplan, S. (1988). "Simplifying condition term rewriting systems unification, termination, and confluence," *J. of Symbolic Computation* **4**(3), 295–334.

Kaplan, S., and Rémy, J.-L. (1988). "Completion algorithms for conditional rewriting systems," *Resolution of Equations in Algebraic Structures*. Academic Press, New York.

Kapur, D. (1980). "Towards a theory of abstract data types," Report LCS-TR-237, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA.

Kapur, D., and Musser, D. R. (1986). "Inductive reasoning with incomplete specifications," *Proceedings of the Symposium on Logic in Computer Science*, Cambridge, MA, 367–377.

Kapur, D., and Musser, D. R. (1987). "Proof by consistency," *Artificial Intelligence* **31**(2), 125–377. (Previous version in the *Proceedings of the NSF Workshop on the Rewrite Rule Laboratory*, Schenectady, NY. 245–267, September 1983.)

Kapur, D., and Narendran, P. (1985a). "An equational approach to theorem proving in first-order predicate calculus," *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, Los Angeles, CA, 1146–1153.

Kapur, D., and Narendran, P. (1985b). "A finite Thue system with decidable word problem and without equivalent finite canonical system," *Theoretical Computer Science* **35**, 337–344.

Kapur, D., and Sivakumar, G. (1983). "Experiments with and architecture of RRL, a rewrite rule laboratory," *Proceedings of an NSF Workshop on the Rewrite Rule Laboratory*, Schenectady, NY, 33–56. (Available as Report 84GEN008, General Electric Research and Development, April 1984.)

Kapur, D., Musser, D. R., and Narendran, P. (1988). "Only prime superpositions need be considered for the Knuth–Bendix procedure," *J. Symbolic Computation* **4**, 19–36.

Kapur, D., Narendran, P., and Zhang, H. (1986). "Proof by induction using test sets," *Proceedings of the Eight International Conference on Automated Deduction*, Oxford, England, 99–117. (Available as Vol. 230, *Lecture Notes in Computer Science*, Springer, Berlin.)

Kapur, D., Narendran, P., and Zhang, H. (1987). "On sufficient completeness and related properties of term rewriting systems," *Acta Informatica* **24**(4), 395–416.

Kirchner, C. (1985). "Methodes et outils de conception systematique d'algorithmes d'unification dans les theories equationnelles," Thèse d'Etat, Université de Nancy, Nancy, France.

Kirchner, H. (1984). "A general inductive completion algorithm and application to abstract data types," *Proceedings of the Seventh International Conference on Automated Deduction*, Napa, CA, 282–302. (Available as Vol. 170, *Lecture Notes in Computer Science*, Springer, Berlin.)

Kirchner, H. (1987). "Schematization of infinite sets of rewrite rules: Application to the divergence of completion processes," *Proceedings of the Second International COnference on Rewriting Techniques and Applications*, Bordeaux, France, 180–191. (Available as Vol. 256, *Lecture Notes in Computer Science*, Springer, Berlin.)

Klop, J. W. (1987). "Term rewriting systems: A tutorial," *Bulletin of the European Association for Theoretical Computer Science* **32**, 143–183.

Knuth, D. E., and Bendix, P. B. (1970). "Simple word problems in universal algebras," In *Computational Problems in Abstract Algebra* (J. Leech, ed.). Pergamon Press, Oxford, U.K., 263–297.

Kounalis, E. (1985). "Validation de spécifications algébriques par complétion inductive," Thèse d'Etat, Université de Nancy, Nancy, France.

Kounalis, E., and Rusinowitch, M. (1987). "On word problems in Horn theories," *Proceedings of the First International Workshop on Conditional Rewriting*, Orsay, France, 144–160. (Available as Vol. 308, *Lecture Notes in Computer Science*, Springer, Berlin.)

Kowalski, R. A. (1974). "Predicate logic as programming language," *Proceedings of the IFIP Congress*, Amsterdam, The Netherlands, 569–574.

Küchlin, W. (1986a). "A generalized Knuth–Bendix algorithm," Technical Report 86-01, Department of Mathematics, Swiss Federal Institute of Technology, Zurich, Switzerland.

Küchlin, W. (1986b). "Equational completion by proof transformation," Ph.D. Thesis, Department of Mathematics, Swiss Federal Institute of Technology, Zurich, Switzerland.

Küchlin, W. (1987). "Inductive completion by ground proof transformation," Technical Report 87-08, Department of Computer and Information Sciences, University of Delaware, Newark, DE.

Lankford, D. S. (1975). "Canonical inference," Memo ATP-32, Automatic Theorem Proving Project, University of Texas, Austin, TX.

Lankford, D. S. (1981). "A simple explanation of inductionless induction," Memo MTP-14, Department of Mathematics, Louisiana Tech. University, Ruston, LA.

Lankford, D. S., and Ballantyne, A. M. (1977a). "Decision procedures for simple equational theories with commutative axioms: Complete sets of commutative reductions," Memo ATP-35, Department of Mathematics and Computer sciences, University of Texas, Austin, TX.

Lankford, D. S., and Ballantyne, A. M. (1977b). "Decision procedures for simple equational theories with commutative-associative axioms: Complete sets of commutative-associative reductions," Memo ATP-39, Department of Mathematics and Computer Sciences, University of Texas, Austin, TX.

Lankford, D. S., Ballantyne, A. M. (1979). "The refutation completeness of blocked permutative narrowing and resolution." *Proceedings of the Fourth Workshop on Automated Deduction*, Austin, TX, 53–59.

Lankford, D. S., and Ballantyne, A. M. (1983). "On the uniqueness of term rewriting systems," Unpublished note, Department of Mathematics, Louisiana Tech. University, Ruston, LA.

LeChenadec, P. (1985). *Canonical Forms in Finitely Presented Algebras*. Pitman-Wiley, London.

Lescanne, P. (1983). "Computer experiments with the REVE term rewriting system generator," *Proceedings of the Tenth ACM Symposium on Principles of Programming Languages*, Austin, TX, 99–108.

Lescanne, P. (1984). "Term rewriting systems and algebra," *Proceedings of the Seventh International Conference on Automated Deduction*, Napa, CA, 166–174. (Available as Vol. 170, *Lecture Notes in Computer Science*, Springer, Berlin.)

Lescanne, P. (1986). "Divergence of the Knuth–Bendix procedure and termination orderings," *Bulletin of the European Association for Theoretical Computer Science* **30**, 80–84.

Livesey, M., and Siekmann, J. (1976). "Unification of A+C-terms (bags) and A+C+I-terms (sets)," Intern. Ber. Nr. 5/76, Inst. für Informatik, University Karlsruhe, Karlsruhe, West Germany.

Manna, Z. (1974). *Mathematical Theory of Computation*. McGraw-Hill, New York.

Manna, Z., and Waldinger, R. J. (1980). "A deductive approach to program synthesis," *ACM Transactions on Programming Languages and Systems* **2**(1), 90–121.

Marcus, L. (1984). "Demons and equivalence," Technical Report ATR-83(8478)-4, Computer Science Laboratory, The Aerospace Corporation, El Segundo, CA.

Martin, U. (1987). "How to choose the weights in the Knuth–Bendix ordering," *Proceedings of the Second International Conference on Rewriting Techniques and Applications*, Bordeaux, France, 42–53. (Available as Vol. 256, *Lecture Notes in Computer Science*, Springer, Berlin.)

Metivier, Y. (1983). "About the rewriting systems produced by the Knuth–Bendix completion algorithm," *Information Processing Letters* **16**(1), 31–34.

Musser, D. R. (1980). "On proving inductive properties of abstract data types," *Proceedings of the Seventh ACM Symposium on Principles of Programming Languages*, Las Vegas, NV, 154–162.

Nelson, C. G., and Oppen, D. C. (1980). "Fast decision procedures based on congruence closure," *J. of the Association for Computing Machinery* **27**(2), 356–364. (Previous version appeared as "Fast decision algorithms based on union and find," *Proceedings of the Symposium on Foundations of Computer Science*, Stanford, CA, 114–119, 1977.)

Newman, M. H. A. (1942). "On theories with a combinatorial definition of 'equivalence'," *Annals of Mathematics* **43**(2), 223–243.

Nipkow, T., and Weikum, G. (1982). "A decidability result about sufficient-completeness of axiomatically specified abstract data types," *Proceedings of the Sixth GI Conference on Theoretical Computer Science*, 257–268. (Available as Vol. 145, *Lecture Notes in Computer Science*, Springer, Berlin.)

Padawitz, P. (1983). "Correctness, completeness, and consistency of equational data type specifications," Bericht Nr. 83-15, Technische Univerität, Berlin, West Germany.

Paul, E. (1984). "Proof by induction in equational theories with relations between constructors," *Proceedings of the Ninth Colloquium on Trees in Algebra and Programming*, Cambridge University Press, Bordeaux, France, 211–225.

Paul, E. (1985). "Equational methods in first order predicate calculus," *J. Symbolic Computation* **1**(1), 7–29. (Previous version appeared as "A new interpre-

tation of the resolution principle," *Proceedings of the Seventh International Conference on Automated Deduction*, Napa, CA, 333–335, May 1984.)

Paul, E. (1986). "On solving the equality problem in theories defined by Horn clauses," *Theoretical Computer Science* **44**(2), 127–153. (Previous version in *Proceedings of the European Conference on Computer Algebra: Research Contributions*, Linz, Austria, 363–377, April 1985.)

Pedersen, J. (1985). "Obtaining complete sets of reductions and equations without using special unification algorithms," *Proceedings of the European Conference on Computer Algebra: Research Contributions*, Linz, Austria, 422–423. (Available as Vol. 204, *Lecture Notes in Computer Science*, Springer, Berlin.)

Peterson, G. E. (1983). "A technique for establishing completeness results in theorem proving with equality," *SIAM J. on Computing* **12**(1), 82–100. (Previous version in *Proceedings of the First Annual National Conference on Artificial Intelligence*, Stanford, CA, 87–89, August 1980.)

Peterson. G. E., Stickel, M. E. (1981). "Complete sets of reductions for some equational theories," *J. of the Association for Computing Machinery* **28**(2), 233–264.

Plaisted, D. A. (1979), personal communication.

Plaisted. D. A. (1980). "Partial correctness and semantic confluence of term-rewriting systems," Unpublished manuscript, Department of Computer Science, University of Illinois, Urbana, IL.

Plaisted, D. A. (1982). "An initial algebra semantics for error presentations," Unpublished report, Computer Science Laboratory, SRI International, Menlo Park, CA.

Plaisted, D. A. (1985a). "The undecidability of self embedding for term rewriting systems," *Information Processing Letters* **20**(2), 61–64.

Plaisted, D. A. (1985b). "Semantic confluence tests and completion methods," *Information and Control* **65**(2/3), 182–215.

Plaisted, D. A. (1986a). "A simple non-termination test for the Knuth-Bendix method," *Proceedings of the Eighth International Conference on Automated Deduction*, Oxford, England, 79–88. (Available as Vol. 230, *Lecture Notes in Computer Science*, Springer, Berlin.)

Pottier, L. (1987). "Limitation des paires critiques dans les complétion de systèmes de réécriture formelle," Rapport, Laboratoire d'Informatique, Université de Nice, Nice, France.

Purdom, P. (1987). "Detecting loop simplifications," *Proceedings of the Second International Conference on Rewriting Techniques and Applications*, Bordeaux, France, 54–61. (Available as Vol. 256, *Lecture Notes in Computer Science*, Springer, Berlin.)

Reddy, U. S. (1985). "Narrowing as the operational semantics of functional languages," *Proceedings of the IEEE Symposium on Logic Programming*, Boston, MA, 138–151.

Rémy, J.-L. (1982). "Etude des systèmes de réécriture conditionnels et applications aux types abstraits algébriques," Thèse, Institut National Polytechnique de Lorraine, Nancy, France.

Rémy, J.-L., and Zhang, H. (1984). "REVEUR4: A system for validating conditional algebraic specifications of abstrac data types," *Proceedings of the Sixth European Conference on Artificial Intelligence*, Pisa, Italy, 563–572.

Réty, P., Kirchner, C., Kirchner, H., and Lescanne, P. (1985). "NARROWER: A new algorithm for unification and its application to logic programming," *Proceedings of the First International Conference on Rewriting Techniques*

and Applications, Dijon, France, 141–157. (Available as Vol. 202, *Lecture Notes in Computer Science*, Springer, Berlin, September 1985.)

Robinson, G., and Wos, L. (1969). "Paramodulation and theorem-proving in first order theories with equality," *Machine Intelligence 4*, Edinburgh University Press, Edinburgh, Scotland, 135–150.

Rusinowitch, M. (1987). "Theorem proving with resolution and superposition: An extension of Knuth & Bendix procedure as a complete set of inference rules," Internal Report 87-R-128, Centre de Recherche en Informatique de Nancy, Nancy, France.

Siekmann, J. (1984). "Universal unification," *Proceedings of the Seventh International Conference on Automated Deduction*, Napa, CA, 1–42. (Available as Vol. 170, *Lecture Notes in Computer Science*, Springer, Berlin.)

Siekmann, J., and Szabo, P. (1982). "A Noetherian and confluent rewrite system for idempotent semigroups," *Semigroup Forum* **25**, 83–110.

Slagle, J. R. (1974). "Automated theorem-proving for theories with simplifiers, commutativity, and associativity," *J. of the Association for Computing Machinery* **21**(4), 622–642.

Stickel, M. E. (1981). "A unification algorithm for associative-commutative functions," *J. of the Association for Computing Machinery* **28**(3), 423–434.

Stone, M. (1936). "The theory of representations for Boolean algebra," *Transactions of the American Mathematical Society* **40**, 37–111.

Takeuti, G. (1987). *Proof Theory*. North-Holland. (Revised edition.)

Thiel, J. J. (1984). "Stop losing sleep over incomplete data type specifications," *Proceedings of the Eleventh ACM Symposium on Principles of Programming Languages*, Salt Lake City, UT, 76–82.

Thomas, C. (1984). "RRLab—Rewrite rule labor," Memo SEKI-84-01, Fachbereich Informatik, Universität Kaiserslautern, Kaiserslautern, West Germany.

Watts, D. E., and Cohen, J. K. (1980). "Computer implemented set theory," *American Mathematical Monthly* **87**(7), 577–560.

Winkler, F. (1984). "The Church–Rosser property in computer algebra and special theorem proving: An investigation of critical-pair/completion algorithms," Thesis, University of Linz, Linz, Austria.

Winkler, F. (1985). "Reducing the complexity of the Knuth–Bendix completion algorithm: A 'unification' of different approaches," *Proceedings of the European Conference on Computer Algebra: Research Contributions*, Linz, Austria, 378–389. (Available as Vol. 204, *Lecture Notes in Computer Science*, Springer, Berlin.)

Winkler, F., and Buchberger, B. (1983). "A criterion for eliminating unnecessary reductions in the Knuth–Bendix algorithm," *Proceedings of the Colloquium on Algebra, Combinatorics and Logic in Computer Science*, Györ, Hungary.

Zhegalkin, I. I. (1927). "On a technique of evaluation of propositions in symbolic logic," *Matematicheskii Sbornik* **34**(1), 9–27.