An Introduction to Knuth–Bendix Completion

A. J. J. DICK

Informatics Department, Systems Engineering Division, Rutherford Appleton Laboratory, Chilton, Didcot, Oxon OX14 4LZ‡

An informal introduction is given to the underlying concepts of term rewriting. Topics covered are Knuth–Bendix completion, completion modulo equations, unfailing completion and theorem proving by completion.

Received August 1990

1. INTRODUCTION

This paper aims to provide an informal and intuitive introduction to many of the concepts essential to a clear understanding of Knuth–Bendix completion. Very few of the ideas expressed in this introductory material are, in essence, original to the author. A similar paper was first published as a research report in 1984,[3] and was later expanded to form part of the author's thesis.[4] The intention is to motivate the key concept of 'superposition' from the standpoint of equational reasoning. The idea of proof complexity is inspired by a PhD thesis by Bachmair.[1] A broad survey of the field of term rewriting can be found in a paper by Huet and Oppen.[14]

For the purposes of the paper, brief, semi-formal definitions of the terminology are given below.

2. TERMINOLOGY

Terms are constructed from unsorted, fixed arity *function symbols* and *variables* in the usual way. For instance if '+', '−' and '0' are binary, unary and nullary function symbols, respectively, and x, y, z are variables, then $0+x$, $(--x+-y)+y$ and $x+(y+z)$ are examples of terms. Terms will be denoted by t, t', t'', t_1, t_2, ..., etc.

A term t is a *subterm* of itself; and if t is of the form $f(t_1, \ldots t_n)$, where f is a function symbol of arity n, then any subterm of $t_1, \ldots$ or t_n is also a *subterm* of t.

A variable x may be *instantiated* by replacing it with any term. The operation of *substitution*, denoted by $t[t'/x]$, instantiates every occurrence of x in t by t'. Variables will be denoted by x, y, z, u, v, w, a, b, ..., etc.

An *equation* is a pair of terms, $\langle t, t' \rangle$, usually denoted by $t = t'$. A set of equations, S, defines a relation on terms where t is related to t' if and only if $t = t' \in S$. The closure of this relation under symmetry, reflexivity, transitivity, instantiation and subterm replacement is the *equality relation defined by S*.

Given terms t and t', *matching* t to t' is the process of finding substitutions of terms for variables in t that render t identical to t'; *unifying* t *and* t' is the process of finding substitutions of terms for variables in both t and t' that render the two terms identical.

Other terminology will be introduced as needed.

3. EQUATIONAL REASONING

Equations provide a natural framework in which to express abstractly the properties of functions and how they interact in algebraic systems. The ability freely to interchange equal terms in whatever context they may appear gives rise to a very important method of mathematical reasoning – the ability to explore the properties and implications of a set of equational axioms by repeatedly replacing equals for equals. Such a technique is widely reflected in the presentation of algebraic proofs. Figure 1 shows a typical proof of $--a = a$ in group theory defined by left and right identity, left inverse, and associativity.

A close examination of the reasoning process embodied in Fig. 1 reveals that there is a great deal going on behind the scenes of an apparently simple proof. To illustrate just one aspect, consider the application of axiom A3 in step 2, in which 0 is replaced by $-a+a$. At this stage of the proof, the motivation for instantiating the variable x to a is not obvious.* Its purpose is only revealed at step 4, where the second application of A3 requires such a binding. Perhaps it is misleading to represent the proof as a sequence of such steps when, in reality, the author of the proof may have used a quite different intuition to derive it (from step three outwards, for instance). However, a more accurate rendering of the step-wise proof may be as found in Fig. 2.

Such observations suggest that, in considering the automation of such reasoning, the following basic processes are necessary:

Unification

Finding variable instantiations that will unify a term with one side of an axiom. The brief discussion of steps 3 and 4 above showed that variables in both the axiom and the term to which the axiom is being matched must be instantiated. Matching, which will only instantiate variables in the axioms, is therefore not sufficient.

Rewriting

Replacing one side of an axiom by the other side within the context of a term. (Replacing equals by equals.)

Strategy

Finding the sequence of axiom applications that show two terms to be equal. This is where the major difficulty lies. There is an infinite network representing the closure of the equality relation. This search space is ridden with infinite sequences and loops requiring a very cautious

‡ Now at: Bull SA, 68 route de Versailles, 78430 Louveciennes, France

* One can envisage a strategy in which variables are only bound to existing elements of the original terms. Such a technique would, it seems, work satisfactorily in this case, but it does not provide a solution to the binding problem in general.

Given the four axioms

$$0+x = x \quad \text{(A1)}$$
$$x+0 = x \quad \text{(A2)}$$
$$-x+x = 0 \quad \text{(A3)}$$
$$(x+y)+z = x+(y+z), \quad \text{(A4)}$$

prove that $--a = a$ for any a.

Proof :-	1)	$--a$	=	$--a+0$	by A2
	2)		=	$--a+(-a+a)$	by A3
	3)		=	$(--a+-a)+a$	by A4
	4)		=	$0+a$	by A3
	5)		=	a	by A1

Figure 1. A typical proof of $--a = a$.

Proof :-	1)	$--a$	=	$--a+0$	by A2
	2)		=	$--a+(-x+x)$	by A3
	3)		=	$(--a+-x)+x$	by A4
	4)		=	$0+a$	by A3 with x instantiated to a
	5)		=	a	by A1

Figure 2. An alternative representation of the proof of $--a = a$.

approach. At the very least, one must (a) record the path followed through the search space, and avoid repetition; (b) adopt a 'breadth first' strategy which favours 'less complex' terms, to avoid getting lost in ever divergent paths.

A naive strategy such as this one is totally non-deterministic, and requires full backtracking. For anything but the most trivial problems, a vast amount of searching is likely to be required before a proof is found.

The aim of Knuth–Bendix completion is to provide, if possible, a 'deterministic' strategy for solving the equality problem expressed by a given set of axioms. This is only possible in certain cases; in others, Knuth–Bendix completion may provide only a semi-decision procedure. The purpose of this paper is to introduce informally the principles involved.

4. COMPLEXITY OF TERMS

First some important observations are made about the axioms in Fig. 1 which will help in understanding the nature of proofs.

Suppose there exists an ordering on terms, $>_r$, which describes their 'complexity'. For example, in axioms A1, A2 and A3,

$0+x >_r x$ ($0+x$ 'is more complex than' x)
$x+0 >_r x$ ($x+0$ 'is more complex than' x)
$-x+x >_r 0$ ($-x+x$ 'is more complex than' 0).

It is easy to see how 'complexity' could somehow be related to size. The associativity axiom A4, however, is more difficult, since both sides appear to be of exactly the same size. The notion of complexity could be extended, so that, in the case where terms are of the same size, their structure is taken into account. For instance, a term could be regarded as being simpler than another, if, viewing terms as trees, its left-most subtree is less complex than the corresponding subtree in the other term, as illustrated in Fig. 3. Axioms may now be viewed

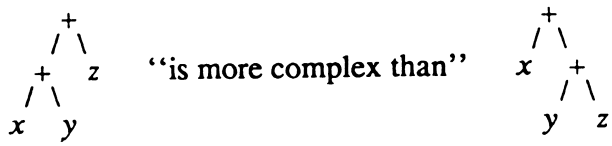

Figure 3. Complexity of terms in the associative axiom.

as simplification rules, or *rewrite rules*, i.e. rules that allow complex terms to be rewritten in some simpler (but equivalent) form. Rules are written as $t \rightarrow t'$, meaning that $t = t'$ and $t >_r t'$. Figure 4 shows the axioms of

$$0+x \rightarrow x \quad \text{(R1)}$$
$$x+0 \rightarrow x \quad \text{(R2)}$$
$$-x+x \rightarrow 0 \quad \text{(R3)}$$
$$(x+y)+z \rightarrow x+(y+z) \quad \text{(R4)}$$

Figure 4. Group axioms as rewrite rules.

Fig. 1 expressed as rewrite rules. The process of repeatedly applying a rewrite rules to a term is known as *reduction*, or *normalisation*. If no rules apply to a term, then it is said to be in *normal form*.

A set of rewrite rules, R, defines a relation on terms where t is related to t' if and only if $t \rightarrow t' \in R$. The closure of this relation under instantiation and subterm replacement is the *reduction relation defined by* R, also denoted by $\rightarrow$.

Requiring that the rewrite rules alone be complexity reducing with respect to some well-founded reduction ordering is not sufficient to ensure that the reduction relation is well-founded. Since the reduction relation is closed under instantiation and subterm replacement, then the reduction ordering must be so too.

For any terms t,t',t'' and variable x, an ordering on terms is said to be *stable* (with respect to term structure) if

$$t' >_r t'' \quad \text{implies that } t'[t/x] >_r t''[t/x],$$

and *monotonic* (with respect to term structure) if

$$t' >_r t'' \quad \text{implies that } t[t'/x] >_r t[t''/x].$$

Any set of rewrite rules, R, in which each rule $l \rightarrow r \in R$ satisfies $l >_r r$ for some well-founded, stable and monotonic ordering on terms, defines a well-founded reduction relation.[2] A set of rewrite rules with this property is said to be *Nœtherian*.

Treating equations as rewrite rules greatly decreases the space of possible applications. Indeed, with a well-founded reduction relation, the space is finite, because there are no infinite rewriting sequences $t \rightarrow t' \rightarrow t'' \rightarrow \ldots$.

There are, of course, many ways of defining such orderings. The reader is referred to the work of Dershowitz[2] for an excellent survey of various methods. The reduction ordering used here and in subsequent sections is the Knuth–Bendix ordering.[20] It works by weighting terms according to the symbols that occur in them, weightier terms being considered as more complex. Where two terms are of equal weight, either operator precedence is used, or subterms are considered lexicographically, having a similar effect on the associativity axiom as shown in Fig. 3.

Because of the nature of reduction orderings, all variables that occur in the right-hand side of a rewrite rule must also occur in the left-hand side. If this were not so, a stable reduction ordering could not be constructed, since the rogue variable on the right-hand side could always be replaced by a term of sufficient complexity to make the ordering unstable.

There is an advantage attached to this restriction on variables: no new variables are introduced into terms as they are rewritten. This means that the kind of instantiation problem discussed in Section 2 does not occur. When applying rules, therefore, variable instantiations need only occur in the rule, and not in the term being reduced, making full unification unnecessary. Matching alone is sufficient, which is a special case of unification, and a much more efficient process. Such a constraint is not at all unreasonable, since it is natural to create rules that eliminate variables rather than introduce new ones. However, there are some cases in which this restriction is undesirable, and ways of overcoming this limitation will be discussed later. Note that rules R1–R4 do satisfy the constraint.

Figure 5 shows the proof of Fig. 1 arranged in a way that demonstrates changes in term complexity. Positioning of terms is in approximate accordance with their relative complexity, the more complex terms being higher

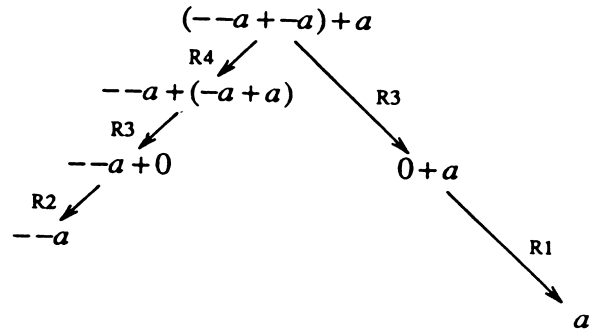

Figure 5. Term complexity in the proof of $- -a = a$.

on the diagram. Viewed in this light, the whole essence of the proof seems to be the top, most complex term $(- -a+-a)+a$. Starting from $- -a$, the complexity of the terms builds up to a peak, and is then simplified in another direction. A proof that does not contain a peak term is called a *rewrite proof*, because both sides of the theorem reduce to the same normal form by simple application of the rewrite rules. All non-rewrite proofs contain at least one peak term, called a *critical term*, which can be rewritten in two different ways. Note that the theorem being proved is represented by terms in normal form, $- -a$ and a. Figures 6 and 7 give other examples. In the first is a rewrite proof, and in the second, two critical terms are involved in a non-rewrite proof.

Given the rules R1 to R4,
prove that $(-a+a)+a = (a+-b)+b$ for all a and b.

Proof :-

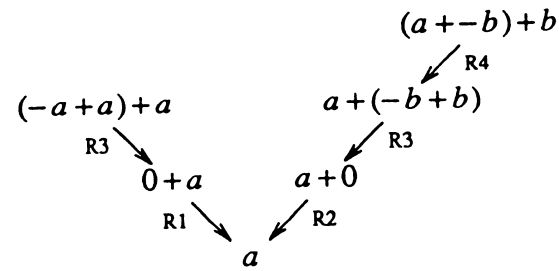

Figure 6. An example of a rewrite proof.

The 'eureka' step of devising a proof would seem to be the discovery of one or more critical terms, from which proofs could be constructed by reducing these terms to alternative normal forms.

How can these critical terms be generated? For a term to be rewritten in two different ways there must be two rules that apply to it (or one rule that applies in two distinct ways). In other words, a critical term must contain two separate instances of left-hand sides of rewrite rules. Such terms could perhaps be found, then, by the unification of the left-hand sides of rules. This idea is treated in the following section.

5. SUBSUMPTION, UNIFICATION AND SUPERPOSITION

If a term t' can be formed from another term t by making

Given axioms R1, R3 and R4,
prove axiom R2, i.e. that $a+0=a$ for all a. (See Figure 4.4).

Proof:-

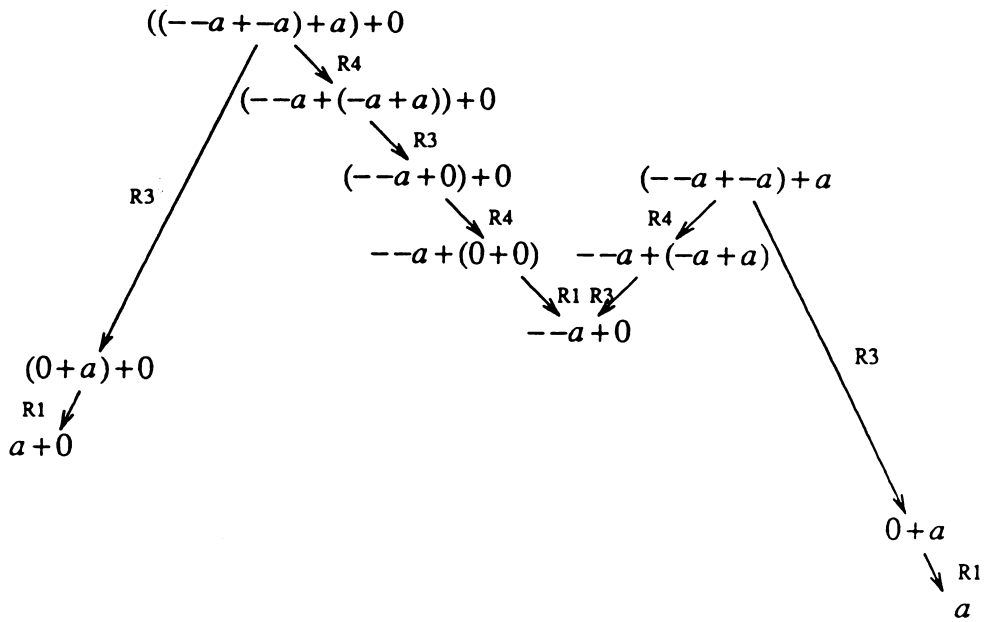

Figure 7. An example of a non-rewrite proof with two peaks.

substitutions for the variables of t, then we say that t' is an instance of t. In other words, t is more general then t', or t subsumes t'. This forms the basis of a relation on terms called the subsumption ordering, written $t' \geqslant_t t$, meaning that t' is an instance of t.

The subsumption relation is closely related to the ability to match one term to another. For instance, if $t = t'[t''/x]$, then the substitution $[t''/x]$ matches t' to t. If $t \geqslant_t t'$ then there exists a substitution which matches t' to t. If $t \geqslant_t t'$ and $t' \geqslant_t t$, then t and t' differ only by the names of their variables. The matching substitutions in this case do no more than rename the variables. Thus the subsumption relation is in fact a pre-ordering, since it is only anti-symmetric upto the renaming of variables. If neither $t \geqslant_t t'$ nor $t' \geqslant_t t$, then t and t' cannot be matched, and are said to be unrelated.

Let t^* stand for the set of all instances of a term t. In general, t^* is infinite, and is represented diagrammatically in Fig. 8.

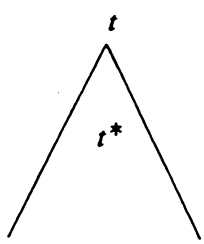

Figure 8. Representation of t^*, the set of a term t.

The two sets of instances t_1^* and t_2^*, of terms t_1 and t_2, respectively, may intersect as represented in Fig. 9. Although t_1 and t_2 may themselves be unrelated, there will often be instances of each that *are* related. These lie in $t_1^* \bigcap t_2^*$, the set of terms that are instances of both t_1 and t_2.

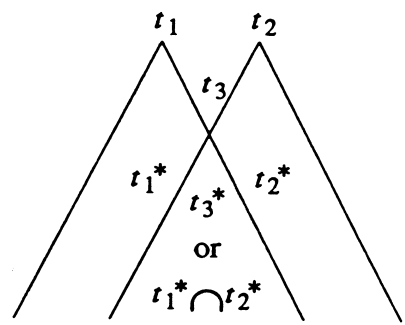

Figure 9. Representation of the intersection of t_1^* and t_2^*.

The unification of t_1 and t_2 is the process of finding elements of $t_1^* \bigcap t_2^*$. If t_1 and t_2 are the left-hand sides of rewrite rules with distinct variables,† then the set of unified forms of t_1 and t_2 is a (usually infinite) set of critical terms to which both rules can be applied.

Simple unification, however, is not sufficient to find all critical terms, because a rule may be applied to any part of a term, not just the whole of it. For this reason, the left-hand side of each rule must be unified with all possible subterms‡ of left-hand sides.

For example, the critical term $(- \ -a+-a)+a$ in the proof of $- \ -a = a$ in Fig. 5 can be rewritten by rule R3

† If t_1 and t_2 share variables, then their set of unified forms is, in general, a subset of, and not identical to, $t_1^* \bigcap t_2^*$. As an example of this, consider the two terms $f(x, 2)$ and $f(3, x)$: an instance of both is $f(3, 2)$, but this is not a unified form. Since the variables occurring in all rewrite rules are universally quantified, variable conflict can always be avoided by renaming the variables in one of the rules.

‡ In practice, a subterm that is a simple variable is not unified, since this yields a critical term of no practical value. The two instances must overlap in the superposed form (see the Knuth–Bendix paper,[20] Section 5).

and rule R4, and therefore contains instances of the left-hand sides of both these rules. Discovery of this critical term would necessitate the unification of the subterm $(x+y)$ of the left-hand side of R4, with the whole of the left-hand side of R3, $(-x'+x')$ (variables have been renamed to avoid confusion). A possible substitution is $[- -a/x][-a/y][a/z][-a/x']$ giving $- -a+-a$ when applied to the left-hand side of R3, and the critical term $(- -a+-a)+a$ when applied to the left-hand side of R4.

The critical term $(- -a+-a)+a$ may now be rewritten by the two rules used in its formation, giving two new terms called a *critical pair*:

$$\begin{array}{ccc} & (--a+-a)+a & \\ \text{R4}\swarrow & & \searrow\text{R3} \\ --a+(-a+a) & & 0+a \end{array}$$

The process of finding a critical pair by unifying subterms of the left-hand sides of two rules is called *superposition*.

If superposition is proposed as a means of discovering critical terms, the question arises as to which superpositions should be selected. Generating and processing an infinite set of critical terms does not seem to offer any advantage over the naive strategy suggested in Section 3.

Fortunately, it is a well-known fact, dating from the work of Robinson,[28] that unification of first-order terms is unitary; that is, if the unification of two first-order terms is possible, it yields a single, most general unified form, unique to within renaming. This most general unified form is the one that subsumes all other unified forms, and is the term t_3 in Fig. 9.

Superposition being a form of multiple unification likewise yields a finite set of maximal critical terms. So given a finite set of rules, it is possible to generate a finite set of critical terms which subsume all others. Figure 10 shows the set of critical pairs between the rules of Fig. 4.

The normalisation of these critical pairs reveals that all but two of them have rewrite proofs; that is, the normal forms of both terms in each pair are identical, as shown in Fig. 11. The two critical pairs that have no rewrite proof are nonetheless sound with respect to the original axioms, and it is reasonable to orient these equations using the reduction ordering $>_r$, and consider them as new rewrite rules:

$$-0 \rightarrow 0 \qquad \text{(R9)}$$

$$-x+(x+z) \rightarrow z \qquad \text{(R10)}$$

There are several consequences of adding these new rules. Amongst other things, the proof of $- -a = a$ in Fig. 5 can now be simplified. The terms involved in the three-step proof of R10, as given in Figure 11, all subsume the corresponding terms in the last three steps of the proof of $- -a = a$ in Fig. 5 by the substitutions $[- -a|x]$ and $[a|z]$. Therefore, the last three steps can be replaced by a single step using rule R10, forming the new proof shown in Fig. 12. This proof is obviously less complex, in the sense that it has fewer steps. It still has a peak, and thus is not a rewrite proof.

Another consequence of introducing new rules, is that new superpositions are possible. They are summarised in Fig. 13. Simplification of these new critical pairs allows the derivation of more theorems, including the one shown in Fig. 14.

Again, this proof subsumes that of Fig. 12 by the substitution $[a/x]$, and so including this as a new rule

$$- -x \rightarrow x \qquad \text{(R11)}$$

allows the proof of $- -a = a$ to be simplified to a single application of rule R11, as shown in Fig. 15. It is now a rewrite proof.

It is not hard to see that, if this process is repeated for long enough, any proof can be simplified into a rewrite

$$\begin{array}{ccc} & 0+0 & \\ \text{R1}\swarrow & & \searrow\text{R2} \\ 0 & & 0 \end{array} \qquad \begin{array}{ccc} & -0+0 & \\ \text{R2}\swarrow & & \searrow\text{R3} \\ -0 & & 0 \end{array}$$

$$\begin{array}{ccc} & (0+y)+z & \\ \text{R1}\swarrow & & \searrow\text{R4} \\ y+z & & 0+(y+z) \end{array} \qquad \begin{array}{ccc} & (x+0)+z & \\ \text{R2}\swarrow & & \searrow\text{R4} \\ x+z & & x+(0+z) \end{array}$$

$$\begin{array}{ccc} & (x+y)+0 & \\ \text{R2}\swarrow & & \searrow\text{R4} \\ x+y & & x+(y+0) \end{array} \qquad \begin{array}{ccc} & (-x+x)+z & \\ \text{R4}\swarrow & & \searrow\text{R3} \\ -x+(x+z) & & 0+z \end{array}$$

$$\begin{array}{ccc} & ((x+y)+z)+u & \\ \text{R4}\swarrow & & \searrow\text{R4} \\ (x+(y+z))+u & & (x+y)+(z+u) \end{array}$$

Figure 10. Superpositions possible from rules in Fig. 4.

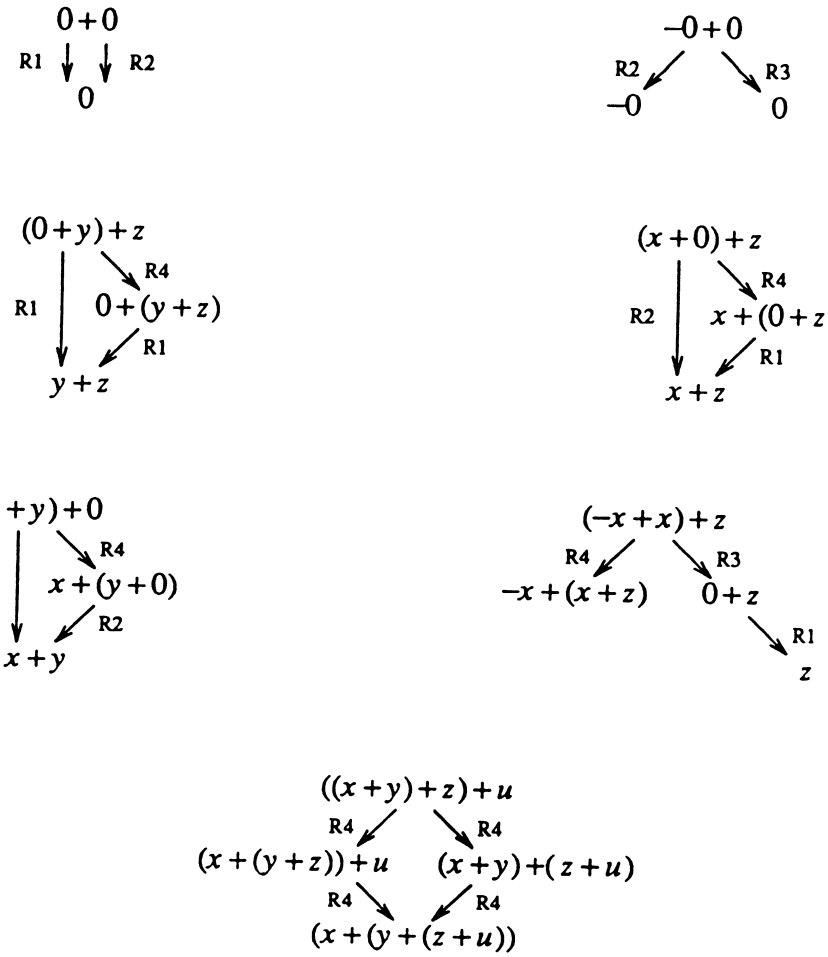

Figure 11. Normalised critical pairs from Fig. 10.

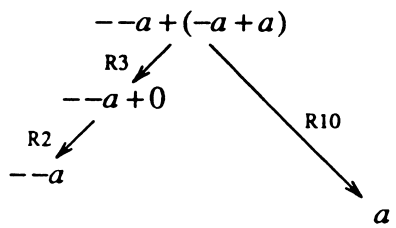

Figure 12. Simplification of the proof of $- -a = a$.

proof. Indeed, if the process is driven by the need to simplify a particular proof, then it is very efficient, since the required superpositions can be deduced from a simple examination of the critical terms occurring in the proof. In this way, the rewrite proof above can be derived after 2 successful superpositions, and 6 successful rule applications.

However, a far more interesting problem is the discovery of a proof. In this case, all superpositions must be considered at every stage, until enough rules have been generated to form a rewrite proof. To discover the proof above, for instance, it may have taken 12 successful superpositions, and 10 successful rule applications. Ensuing sections elaborate on the latter process.

6. CONFLUENCE

What exactly is happening when new rewrite rules derived from critical pairs are introduced? The decreasing complexity of proofs has already been discussed;

$-0 = 0$	from	R3	and	R9
$-0+z = z$		R1		R10
$-x+x = 0+0$		R2		R10
$--x+0 = x$		R3		R10
$-(x+y)+(x+(y+z)) = z$		R4		R10
$0+z = z$		R9		R10
$x+y = --x+y$		R10		R10.

Figure 13. Superpositions possible from the addition of new rules.

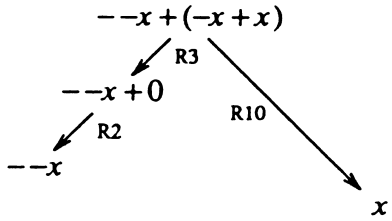

Figure 14. Superposition of rules R10 and R3.

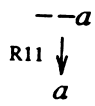

Figure 15. A rewrite proof of $--a = a$.

eventually all proofs will become rewrite proofs. This is because the repertoire of rewrite rules is being extended by others which are equational consequences of the existing rules, thus counteracting the loss of completeness suffered in using rules unidirectionally, and producing a more powerful simplifying engine.

Consider for a moment the critical term $((x+y)+z)+u$ derived from unifying the left-hand side of R4 in Fig. 4 on itself. Its critical pair has a rewrite proof, as shown in Fig. 11. For this reason, there is no point in trying to form a new rule as a result of this superposition; it would not increase the 'power' of the set of rules. Now consider the critical term $-0+0$ derived from rules R2 and R3, with critical pair $\langle -0, 0\rangle$. This critical pair had no rewrite proof, and the new rule R9, $-0 \rightarrow 0$, was formed. When R9 was superposed on R3, the same critical term was derived. On processing $\langle -0, 0\rangle$ the second time, the presence of R9 gave the critical pair a rewrite rule proof, as shown in Figure 16, thus demonstrating the increasing 'power' of the set of rules.

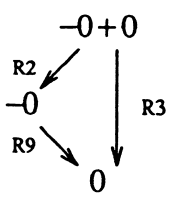

Figure 16. A critical pair with a rewrite proof.

If all critical pairs have rewrite proofs, no new rules need to be produced, and the existing set of rules is said to be *locally confluent*.

In general, a set of rewrite rules is *confluent*, if, whenever a term t can be rewritten to different forms, t_1 and t_2, then t_1 and t_2 have a rewrite proof. Figure 17 illustrates the concept of confluence, often referred to as a 'diamond lemma' from the shape of the diagram. In these figures, $\rightarrow$ is the reduction relation, and $\rightarrow^*$ its transitive and reflexive closure, i.e. reduction by zero or more rules.

Without loss of generality, we can fill in the detail of Figure 17 to pinpoint the exact divergence of the two rewrite sequences $t \rightarrow^* t_1$ and $t \rightarrow^* t_2$, as shown in Fig. 18. Here t'' is a critical term with critical pair $\langle t'_1, t'_2\rangle$, and it is easy to see that, with Noetherian rewriting systems, confluence is equivalent to local confluence as expressed in Fig. 19. This was first proved by Newman.[23]

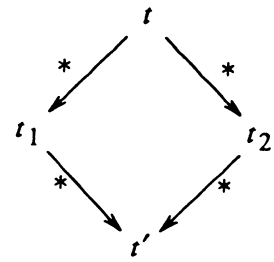

Figure 17. Confluence.

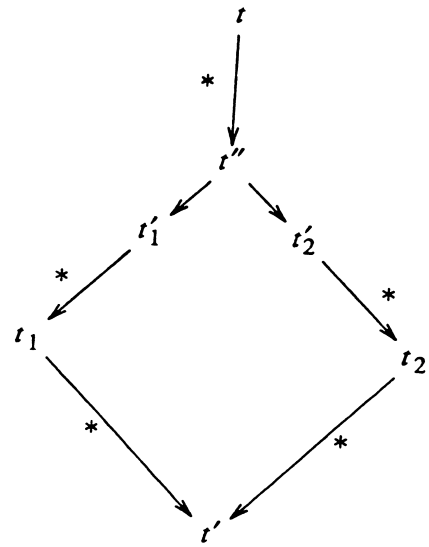

Figure 18. Detail of confluence.

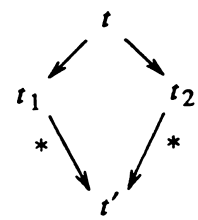

Figure 19. Local confluence.

A key theorem form the Knuth–Bendix paper,[20] known as the *critical pairs lemma*, showed that, to test for local confluence, it is sufficient to consider only those critical terms found by the superposition of the left-hand sides of the rules at non-variable occurrences. The reason for this is clear from the observation that superposition yields terms of minimum interaction between rules, and that, as observed in Section 5, all other critical terms are subsumed by them.

A Noetherian and confluent set of rules is said to be *canonical*, in the sense that every term has a unique normal form, known as its canonical representation; and *Church-Rosser*, in the sense that any two terms, t_1 and t_2, are equal if and only if t_1 and t_2 have a rewrite proof. Therefore, if the set of rewrite rules is Noetherian and locally confluent, the validity of any theorem $t_1 = t_2$ can be tested by reducing t_1 and t_2 to normal form. If the normal forms are the same, the theorem holds; if different, the theorem is proven false. Thus a canonical set of rules provides a decision procedure for reasoning in an equational theory. What is more, the normal forms of terms can be found by applying the rules in any order.

(The data consists of :-

the *axiom set*, a set of equations initially containing the given axioms; and

the *rule set*, an initially empty set of rewrite rules).

while the axiom set is not empty **do**

begin Select and remove an axiom from the axiom set;

Normalise the selected axiom;

if the normalised axiom is not of the form $x=x$ **then**
begin

Order the axiom using the reduction ordering, $>_r$,
and introduce it as a new rule in the rule set;

Superpose the new rule on all existing rules (including itself)
and introduce each critical pair into the axiom set;

end

end.

Figure 20. A simple form of Knuth–Bendix algorithm.

7. KNUTH–BENDIX COMPLETION

Many variations of the Knuth–Bendix completion algorithm are possible. They all use the method of superposition described in Section 5 to create new rewrite rules from a given set of rules or axioms. Figure 20 describes a very simple form of Knuth–Bendix algorithm. The algorithm may behave in one of the following ways:

Converge

The algorithm stops having found a finite canonical set of rewrite rules.

Fail

The algorithm stops because a particular axiom cannot be ordered by $>_r$. For example, no well-founded reduction ordering can orient the commutativity axiom, $a+b=b+a$. If such an axiom is generated, the algorithm (in the form above, at least) cannot continue.

Diverge

Many canonical sets are infinite, and in attempting to complete them, the algorithm never terminates. As shall be described in Section 9, divergent completion algorithms may be used as semi-decision procedures, but full decision procedures are only found when the algorithm converges.

A usual enhancement to the algorithm is to ensure that all rewrite rules are normalised with respect to each other. To achieve this, a normalisation step may be placed in the algorithm before the superposition step, to apply the new rule to the existing rule set. Rules that are reduced only on the right-hand side are unaffected in orientation, or in the way they superpose, and thus may remain as rewrite rules. If, however, a rule is reduced on its left-hand side, it may need to be re-oriented, and new superpositions will have to be considered; for this purpose, such rules are removed from the set or rewrite rules, and become equations. If the introduction of new rules causes existing rules to disappear, the set of rules does not grow consistently with every iteration. Rules that never disappear, and thus form part of the final confluent set, are called *persisting* rules. In the algorithm of Figure 20, all rules persist.

Efficiency is affected considerably by the order in which (a) axioms are selected from the axiom set for the formation of a new rule; and (b) subterms of the left-hand sides of rules are selected for superposition.

In the algorithm portrayed above, the two considerations (a) and (b) are closely linked: as an axiom is selected and formed into a rule, all possible superpositions between the new rule and existing ones are considered. In other forms of the Knuth–Bendix algorithm as given by Huet,[13] for example, these two considerations are kept quite separate. The most flexible arrangement would be to have complete freedom, not only in choosing which axioms to consider as the next rule, but also which particular subterms of left-hand sides of rules to superpose next; and to be able to freely interleave single superpositions with axiom selections.

The particular method an algorithm uses in regard to these considerations is called its *selection strategy*. For Knuth–Bendix completion to be 'complete', its selection strategy must satisfy certain *fairness* constraints. In relation to the considerations above, the constraints are (a) every axiom must eventually be considered as a rule (no axiom can be ignored indefinitely); and (b) every possible superposition of persisting rules must be considered.

Figures 21 and 22 show the operation of a Knuth–Bendix algorithm on two different sets of axioms. The first example is initiated with the group axioms: left

Given axioms :-

A1: $0+x=x$
A2: $-x+x=0$
A3: $(x+y)+z=x+(y+z)$

Operation of the algorithm:-

Derived rules :-	Derivation :- Critical term	From	Rules applied to LHS	RHS
R1: $0+x \to x$		A1		
R2: $-x+x \to 0$		A2		
R3: $(x+y)+z \to x+(y+z)$		A3		
R4: $-x+(x+y) \to y$	$(-x+x)+y$	R3 R2	R3	R2,R1
R5: $-0+x \to x$	$-0+(0+x)$	R1 R4	R1	R4
R6: $--x+0 \to x$	$--x+(-x+x)$	R2 R4	R2	R4
R7: $--x+y \to x+y$	$--x+(-x+(x+y))$	R4 R4	R4	R4
R8: $x+0 \to x$	$--x+(-x+x)$	R2 R4	R2,R7	R4
R9: $-0 \to 0$	$-0+0$	R8 R2	R8	R2
R10: $--x \to x$	$--x+0$	R8 R7	R8	R7,R8
R11: $x+-x \to 0$	$--x+-x$	R7 R2	R7	R2
R12: $x+(-x+y) \to y$	$--x+(-x+y)$	R7 R4	R7	R4
R13: $x+(y+-(x+y)) \to 0$	$(x+y)+-(x+y)$	R3 R11	R3	R11
R14: $x+-(y+x) \to -y$	$-y+(y+(x+-(y+x)))$	R4 R13	R4	R13,R8
R15: $-(x+y) \to -y+-x$	$-y+(y+-(x+y))$	R4 R14	R4	R14

TERMINATES WITH SUCCESS.

Figure 21. Operation of a Knuth–Bendix algorithm on axioms of Fig. 2.

Given axioms :-

A1: $e \cdot x = x$

A2: $x \cdot e = x$

A3: $x \cdot x = e$

A4: $(x \cdot y) \cdot z = x \cdot (y \cdot z)$

Operation of the algorithm:-

Derived rules :-	Derivation :- Critical term	From	Rules applied to:- LHS	RHS
R1: $e \cdot x \rightarrow x$		A1		
R2: $x \cdot e \rightarrow x$		A2		
R3: $x \cdot x \rightarrow e$		A3		
R4: $(x \cdot y) \cdot z \rightarrow x \cdot (y \cdot z)$		A4		
R5: $x \cdot (x \cdot y) \rightarrow y$	$(x \cdot x) \cdot y$	R4 R3	R4	R3,R1
R6: $x \cdot (y \cdot (x \cdot y)) \rightarrow e$	$(x \cdot y) \cdot (x \cdot y)$	R4 R3	R4	R3
R7: $y \cdot (x \cdot y) \rightarrow x$	$x \cdot (x \cdot (y \cdot (x \cdot y)))$	R5 R6	R5	R6,R2
R8: $y \cdot x \; ?{=}? \; x \cdot y$	$x \cdot (x \cdot (y \cdot x))$	R5 R7	R5	R7
TERMINATES WITH FAILURE				

Figure 22. Operation of a Knuth–Bendix algorithm on a commutative group.

identity, left inverse, and associativity. In this case, the completion algorithm converges. The second is a group in which every element is of order two, which is commutative; the completion algorithm generates the commutativity axiom, and terminates with failure when that axiom is considered as a rewrite rule. The algorithm of Fig. 20 is used, but with the rule normalisation step described above, and so not all rules persist. The strategy used to select axioms is by size; the simplest or shortest axiom in the current axiom set is selected first. This strategy is fair and nearly always leads to convergence in fewer iterations than selection on a first-come first-served basis.

In Figs 21 and 22, the generated rewrite rules are listed on the left, with details of their derivations in the remaining columns. Entries in the 'Critical term' column indicate that the rule was derived by superposition; the rules used in the superposition are then listed in the 'From' column. Entries under 'Rules applied to:' indicate rules that have been used to normalise the critical pair. For example, rule R4 in Fig. 21 is derived as shown in Fig. 23.

Of the 16 rules generated in the completion process portrayed in Fig. 21, only 10 persist. Rules R5, R7, R13 and R14 are reduced to trivial identities by subsequent rules. In the case of R6, its left-hand side is reduced, and the resulting equation is formed into R8.

In Fig. 22, the rule R6 is reduced by R7. At the end of the figure, it is found that $y \cdot x$ and $x \cdot y$ cannot be ordered, and the algorithm terminates with failure. It has been proved, however, that the group is commutative.

The implementation used to generate the data for these examples was the default configuration of the ERIL system, described in detail by Dick and Kalmus.[5]

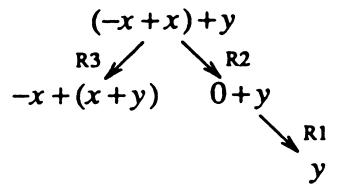

Figure 23. Derivation of rule R4 in Fig. 21.

8. LIMITATIONS OF KNUTH–BENDIX COMPLETION

The limitations of the basic completion algorithm are in four main areas:

(i) An axiom, $l = r$, can only be formed into rewrite rule if $l >_r r$. In practice, it may be very hard to find a reduction ordering to suit particular needs. It would be very useful to be able to reason with unorientable axioms.

(ii) Some axioms, such as commutativity, are by their very nature non-Noetherian. For example, either side of the axiom $x+y=y+x$ can be matched with the other, and whichever way the rule is first applied, it is immediately applicable again, thus generating an infinite rewriting sequence. Special treatment is required for such axioms.

(iii) In many cases of practical importance, the confluent set of rules is infinite, and the algorithm diverges. Frequently, it is apparent that the infinite set consists of sequences of related rules. Finite representations of infinite sequences would be very advantageous.

(iv) Algebraic structures involving partial functions require some form of conditional axiom to prevent the generation of meaningless terms. For example, Fig. 24 shows a critical pair that may be generated from axioms describing part of a division ring. The critical term $0^{-1}.0$ is found, which contains, in effect, a division by zero. The critical pair, $1=0$, causes the equational theory to collapse.

Recent work has attempted to meet some of these problems in a variety of ways. Firstly, by dividing axioms into a set of rewrite rules, R, and a set of equations, E, the notions of rewriting and unification may be generalised to operate on an E-congruence. The first published results were by Lankford and Ballantyne[22] where E consists of permutative axioms. These are equations between terms which differ only under the permutation of symbols. The size of the congruence classes of terms generated by permutative axioms are finite.

This was later generalised by Peterson and Stickel,[26] who define critical pairs modulo finitary unification algorithms (e.g. commutative/associative unification); by this means, distributive lattices and Boolean algebras, for example, can be completed finitely. Huet[12] relaxed some of the restrictions on the nature of the equations in E, but requires rules in R to be left-linear, which means that no variable is permitted to occur more than once in the left-hand side of the rule. Then Jouannoud and Kirchner,[15,16] removed the restriction of left-linearity by further generalisation; the E-congruence classes, however, are still required to be finite. Finally, Bachmair[1] studies the case of rewriting modulo infinite E-congruence classes.

These extensions to completion attempt to treat the permutative axioms. There are other kinds of axiom that cannot be oriented. For instance, there are axioms which do not qualify as rewrite rules, because variables occur in each side that do not occur in the other. An example of this kind can be found in the axioms of an entropic groupoids: $(x\cdot y)\cdot z=(x\cdot w)\cdot z$. The occurrences of y and w prevent the axiom from being oriented in either direction. There is a general method of overcoming the failure case in Knuth–Bendix completion. First published Hsiang and Rusinowitch,[11] it has become known as *unfailing completion*. This subject will be addressed in the next section.

Sorted rewriting systems have been studied extensively by Goguen and Meseguer,[7,8,6] Dick[4] and others. In *many-sorted* systems, the set of terms is partitioned into a finite number of subsets, one for each sort. In *order-sorted* systems, the underlying sorts are allowed to intersect, creating a sub-sorting structure. The latter have been used to treat certain classes of partial algebra, like the one described above, which would otherwise require the use of conditional rewrite rules.

Conditional rewriting systems have been studied by Lankford,[21] Remy[27] and Kaplan[18] amongst others, but this largely remains an open field to which researchers are now turning their attention.

The divergence of the Knuth–Bendix completion procedure has been studied by Hermann and Privara[9] and Kirchner.[19] In the latter, meta-rules and meta-variables are proposed as a means of finitely presenting infinite sequences of rules.

9. UNFAILING COMPLETION

An equation can only be oriented when all possible instantiations of it are orientable with respect to a given reduction ordering, $>_r$. The method of unfailing completion is based on the following very simple observation: if an equation itself cannot be oriented, there may be some instantiations of that equation which can be oriented. This idea suggests the following generalisation of rewriting and superposition:

Rewriting

Apply equations in either direction, but only when its instantiated form (after matching) is orientable under $>_r$. In this way, termination of rewriting is ensured.

Superposition

Allow superposition on both sides of equations. Here the instantiated equation need not be orientable after unification, because termination is in no way affected. However, if the superposed side turns out, after instantiation, to be lower in the reduction ordering than

Given the axioms :-

$$x^{-1}\cdot x \rightarrow 1 \qquad \text{(R12)}$$

$$x\cdot 0 \rightarrow 0 \qquad \text{(R13)}$$

the following critical pair may be derived :-

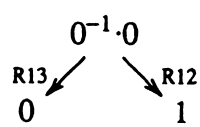

Figure 24. A critical pair that demonstrates a problem with partial operators.

Given the confluent set of rules generated in Figure 21, prove that

$$-((-y+y)+(x+-x)) = x+(-(y+x)+y).$$

The rewrite rules are used to find the normal forms of both terms :-

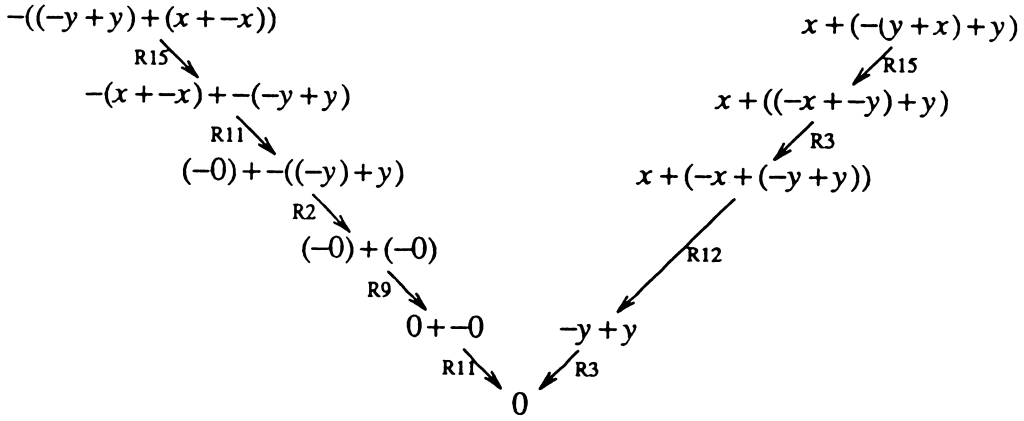

Figure 25. An example proof of validity using a confluent set of rules.

the other side, the critical pair formed will always be trivial.

When an equation can be oriented in its most general form, then ordinary rewriting and superposition fall out as special cases of their unfailing counter-parts. It is obviously more efficient to orient all the equations that are orientable, in order to make use of ordinary rewriting and superposition where ever possible. The less efficient unfailing processes may then be used on the unorientable equations alone.

When performing unfailing completion, both sides of unoriented axioms are matched against any term to be reduced. If a match is successful, an attempt is made to orient the instantiated axiom. Thus the reduction ordering is accessed every time an unorientable axiom has matched a term. For theoretical and practical reasons, the reduction ordering used in this case must be total on ground terms.

Hsiang and Rusinowitch[11] show this method to be complete as a semi-decision procedure for solving the word problem in equational theories. They give some examples which show how finite confluent sets involving unorientable axioms may be generated, where other approaches either fail or diverge.

10. KNUTH–BENDIX COMPLETION AND THEOREM PROVING

In the case where a finite confluent set of rules can be generated, the importance to equational reasoning is clear. A decision procedure is found for solving the identity problem. A two-part strategy for proving theorems is possible.

(1) The given axioms are 'compiled' into a confluent set of rewrite rules using Knuth–Bendix completion. If this succeeds, then:

(2) New equations are shown to be theorems by reducing both sides to normal form. If the normal forms are the same, the theorem is shown to be a consequence of the given axioms; if different, the theorem is proven false.

The complete set of rules in Fig. 21 can be used to prove, for instance, the equation shown in Fig. 25. Since the normal forms are the same, the theorem has been shown to be a consequence of the given axioms after only 9 successful rule applications. Note that, due to confluence, the same normal forms are found regardless of the order of rule application.

Figure 26 shows an attempt to prove a false equation. The diffrent normal forms indicate that the theorem is not a consequence of the given axioms.

In the case where the confluent set is infinite, no such decision procedure is possible. However, the completion process itself may be used a semi-decision procedure. This means that valid equations may be proved by running the completion algorithm long enough to generate the rules required for a rewrite proof; invalid equations, however, cannot be shown invalid. That Knuth–Bendix completion may be used as a semi-decision procedure in this manner was first shown by Huet.[13] Another proof, based on inference rules and proof orderings, is given by Bachmair.[1]

The Knuth–Bendix completion procedure discovers proofs by construction, in the sense that valid consequences of the given axioms are generated until a proof is found. A major problem with this method is that the procedure is not goal directed. The generation of critical pairs is based on the rules formed, and not motivated by the equation to be proved. For this reason, the completion procedure, in many instances, will not provide an efficient proof method. The advantage, however, of constructive proof is that the equational theory is not disturbed by the proving process, and further proofs can be attempted without having to repeat the work already done.

Interesting comparisons have been made between the algebraic completion process and resolution theorem-proving (see, for instance[17]), and recent work by Paul[24] and Hsiang/Dershowitz[10] has proposed ways of using superposition and the completion process to prove theorems in first-order predicate logic. At the heart of these techniques is the realisation that any clause P can

Given the confluent set of rules generated in Figure 21,
prove that $---x = --x + -x$:-

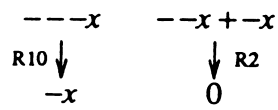

Figure 26. An example of a proof of invalidity using a confluent set of rules.

be made into an equality axiom of the form $P = true$. In cases where user defined functions and equality are embedded in the logic, these methods seem to be considerably more efficient than resolution with paramodulation. Both constructive and destructive proofs are possible in this framework. Destructive proofs are proofs by refutation in which the negation of the desired clause is assumed ($C = false$) and included in the completion process in an attempt to generate a contradiction ($true = false$). By contrast to constructive proof, proof by refutation, in effect, destroys the theory by generating consequences of a false assumption, and every proof must recommence from the start; however, the proof is to a certain extent goal oriented, and experience reported by Paul[25] suggests that contradictions are found very quickly if the theorem to be proved is true. Both constructive and destructive methods are likely to behave in an infinitary manner if the theorem to be proved is false. The author is not aware of published work which explores strategies that are a mixture of the two. Some advantage might be gained, for instance, by partially completing the axioms of the theory before refuting the theorem.

11. CONCLUSIONS

In their original paper, Knuth–Bendix[20] say that their work is 'a precise statement of what hundreds of mathematicians have been doing for many decades'. A good understanding of superposition as a means of selecting 'useful' instantiations of terms represents an important insight into equational reasoning. It is the implicit use of the properties of equality that make the Knuth–Bendix method of superposition eminently more suitable for reasoning about equality than methods that rely on resolution by unification, in which the axioms of equality must be stated explicitly.

Acknowledgements

This paper has grown out of research work funded by the UK Science and engineering Research Council. Thanks are due to members of the ERIL team for their comments, in particular John Kalmus, who has made several very detailed criticisms of the text. Jim Cunningham gave advice in the very early stages. Brian Matthews and Juan Bicarregui have also read various drafts.

REFERENCES

1. L. Bachmair, Proof methods for for equational theories. PhD thesis, University of Illinois (1987).
2. N. Dershowitz, Termination of rewriting. *J. of Symbolic Computation* **3** (1 & 2), 279–301 (1987).
3. A. J. J. Dick, Equational reasoning and the Knuth–Bendix algorithm – an Informal Introduction. Imperial College Research Report DOC 84/21 (Mar. 1984).
4. A. J. J. Dick, Order-sorted equational reasoning and rewrite systems. PhD thesis, Dept. of Computing, Imperial College, London (1988).
5. A. J. J. Dick and J. R. Kalmus, ERIL Users Manual, Version R1.6a. Rutherford Appleton Laboratory, Report RAL-88-055 (Sept. 1988).
6. J. A. Goguen, J-P. Jouannoud and J. Meseguer, Operational semantics for Order-sorted algebra. SRI International, Menlo Park, CA 94305 (1985).
7. J. A. Goguen and J. Meseguer, Completeness of many-sorted equational logic. *SIGPLAN Notices* **16** (7), 24–32 (1981).
8. J. A. Goguen and J. Meseguer, Order-sorted algebra, I: Partial and overloaded operations, errors and inheritance. Draft report, SRI Technical Report, SRI, Menlo Park, CA 94025 (1985).
9. M. Hermann and I. Privara, On nontermination of Knuth–Bendix algorithm. *LNCS* **226**, Springer-Verlag (1986).
10. J. Hsiang and N. Dershowitz, Rewrite methods for clausal and non-clausal theorem proving. In *Proc. 10th ICALP*, pp. 331–346 (July 1983).
11. J. Hsiang and M. Rusinowitch, On word problems in equational theories. Draft Report, Dept. of Computer Science, SUNY at Stony Brook, NY 11794 (Aug. 1986).
12. G. Huet, Confluent reductions: abstract properties and applications to term rewriting systems. *Journal of the ACM* **27** (4), 797–821 (Oct. 1980).
13. G. Huet, A complete proof of correctness of the Knuth–Bendix completion algorithm. *J. of Computer and System Science* **23** (1), 11–21 (Aug. 1981).
14. G. Huet and D. C. Oppen, Equations and rewrite rules – a survey. STAN-CS-80-785 (1980).
15. J-P. Jouannoud, Confluent and coherent equational term rewriting systems: application to proofs in abstract data-types. Procs. of the 8th coll. on *Trees in Algebra and Programming, CAAP'83, LNCS 59*, Springer-Verlag, Berlin (1983).
16. J-P. Jouannoud and H. Kirchner, Completion of a set of rules modulo a set of equations. *SIAM J. of Computing* **15**, 1155–1194 (1985).
17. W. Küchlin, A theorem-proving approach to the Knuth–Bendix completion algorithm. *Lecture Notes in Computer Science* **144** (1983).
18. S. Kaplan, Conditional rewrite rules. *Theoretical Computer Science* (1984).
19. H. Kirchner, Schematization of infinite sets of rewrite rules. Application to the divergence of completion processes. In *Procs. of 2nd Int. Conf. on Rewriting Techniques and Applications, LNCS* **256**, pp. 180–191 Springer-Verlag (1987).
20. D. E. Knuth and P. B. Bendix, Simple word problems in

universal algebras. In *Computational Problems in Abstract Algebra*, ed. J. Leech, pp. 263–297 Pergamon Press (1970).
21. D. S. Lankford, Some new approaches to the theory and applications of conditional term rewriting systems. Rep. MTP-6, Louisiana Tech. Univ., Ruston, Math. Dept. (1979).
22. D. S. Lankford and A. M. Ballantyne, Decision procedures for simple equational theories with permutative axioms: Complete sets of permutative reductions. Rep. ATP-37, Univ. of Texas, Austin: Dep. Math. Comp. Sci. (1977).
23. M. H. A. Newman, On theories with a combinatorial definition of 'Equivalence'. *Annals of Mathematics* **43** (2), 223–243 (1942).
24. E. Paul, Equational methods in first order predicate calculus. *J. Symbolic Computation* **1**, 1–6 (1985).
25. E. Paul, On solving the equality problem in theories defined by horn clauses. *Proc. of EUROCAL'85*, Linz, Austria, Springer-Verlag, *LNCS* **203** (Apr. 1985).
26. G. E. Peterson and M. E. Stickel, Complete sets of reductions for some equational theories. *Journal of the ACM* **28** (2), 233–264 (1981).
27. J-L. Remy, Etude des Systèmes de Reécriture Conditionnels et Applications aux Types Abstraits Algébriques. Thesis, Centre de Recherche en Informatique de Nancy, Nancy, France (July 1982).
28. J. A. Robinson, A machine-oriented logic based on the resolution principle. *Journal of the ACM* **12**, 32–41 (1965).

Book Review

EDWARD R. TUFTE
Envisioning information
Graphic Press, £30.00

When I was first asked to review this book I assumed that the title really meant 'Visualising Information', and basically that is what the book is about. But prior to reading the book I had also assumed that the book's content was concerned with scientific visualisation using computer graphics, but that is not what the book is about. In fact it addresses the graphic design techniques for translating data into information, with examples ranging from sixth-century illuminations on vellum to the three-dimensional distribution of man-made debris about the Earth.

Now having read the book I feel slightly disappointed that the author, Edward Tufte, failed to include any real reference to scientific visualisation using computer graphics, as this is rapidly acquiring a maturity that will make it a major design tool during the next decade. The index does include three references to 'computer visualisations', but no more than two dozen words are given over to this subject, which does not seem adequate, bearing in mind that the author has worked on information design for IBM, Hewlett-Packard, CBS, NBC and the Bureau of the Census.

As one would expect, the book is beautifully illustrated, with only one or two pages that do not contain some form of colour illustration. Within its six chapters on: Escaping Flatland, Micro/Macro Readings, Layering and Separation, Small Multiples, Colour and Information, and Narratives of Space and Time, the reader is presented with various ways of visualising data (my dictionary had no entry for envisioning!), showing ways of depicting eighteenth-century dance steps and Japanese national railroad timetables. But in spite of the excellent illustrations, I was very conscious of the author's descriptive style – it was, to say the least, protracted. For example, page 106 contains the following sentence: 'Above, two rivers meander boustrophedonically around a tight frame, weakening comparison of their lengths'. Now surely there must be a simpler way of expressing this idea without forcing readers to retreat to their dictionaries only to discover that there is no entry for boustrophedonically! I admit that this example is over the top, but I found that I was continually skipping sentences and looking ahead to identify a safe point to recover the current gist.

So what is the book *really* about. Well it contains a hundred, or so, examples of how different graphical approaches have been used to simplify the visual interpretation of multi-dimensional data. The written commentary analyses why the techniques work, but does not offer alternatives which might have improved or hindered the communication process; but to be fair, I do not believe that the author intended it to be a tutorial on the subject, it is simply a collection of effective graphical techniques for communicating the information contained within complex data sets.

To whom is the book directed? Well there is no doubt that students studying graphic design will find it a useful source of how such techniques have evolved historically, but I am not certain how complete the survey is, all that I do know is that the book makes no reference to London's world-famous map of the Underground! But the author writes on page 50: 'Showing complexity is hard work... The conventional economies of declining costs for each additional data bit will usually be offset by a proliferation of elaborate complexities provoked by the interacting graphical elements.'

J. A. VINCE
Rediffusion Simulation Ltd.