ICOT Technical Memorandum: TM-0231

TM-0231

Refutational Theorem Proving for Equational Systems Based on Term Rewriting

by

A. Ohsuga and K. Sakai

September, 1986

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

項書換えに基づく等式システムのための反駁型定理証明

Refutational Theorem Proving for Equational Systems Based on Term Rewriting

大須賀 昭彦　　坂井 公

Akihiko Ohsuga and Kō Sakai

(財) 新世代コンピュータ技術開発機構

Institute for New Generation Computer Technology

あらまし　ICOTでは，知的プログラミングシステム構築活動の一環として，TRSワーキンググループ(TRS-WG)を組織し，プログラミング活動における諸問題の解決に取組んでいる．今回TRS-WGの支援のもとに，TRSに関する種々の技術を研究する目的で，実験システム"Metis" を実装したので，その機能といくつかの実験結果について報告する．

Abstract　The TRS (term rewriting system) Working Group of ICOT has been studying applications of TRSs to the intelligent programming system. As a result, we have implemented a TRS generator called Metis, an experimental tool with the many functions required for such a system. This paper describes the features of Metis and several experiments with it.

1. Introduction

A set of rewrite rules is called a term rewriting system or TRS. The theory of TRSs has a wide variety of both theoretical and practical applications. It provides models for abstract data types, operational semantics for functional programming languages, and inference engines for automated theorem proving with equality.

The intelligent programming system is an important research topic of Japan's Fifth Generation Computer System (FGCS) Project. A lot of evidence suggests that the study of TRSs will yield key technologies for the intelligent programming system, in particular for specification, verification, and synthesis of programs. The Institute for New Generation Computer Technology (ICOT) organized the TRS Working Group in 1985 to study TRSs theoretically, and for application to the intelligent programming system.

Metis is the first result of the activity of the working group. It generates a complete TRS from a set of equations automatically, semi-automatically, or interactively. It is also an experimental tool with the various functions needed for the study of TRSs.

The kernel function of Metis is the so-called Knuth-Bendix completion procedure. Significantly improved with better capabilities and operability by the incorporation of many new facilities. For example, Metis can provide us with several kinds of ordering methods of terms, but the user can orient an equation with little knowledge of the ordering methods and obtain an appropriate rewrite rule that does not violate termination of the TRS. If the equation cannot be oriented to either direction, Metis offers the user several kinds of recipe. It manipulates inequations as well as equations and provides special handling of associative-commutative operators in the completion procedure.

Section 2 describes the basic concept of the TRS. Section 3 introduces the features of Metis in the general framework, and in Section 4, several concrete examples illustrate how Metis actually works.

2. Preliminaries

In this section, we will introduce the terminology and notation in this paper and survey well-known properties of TRSs.

We will deal with finite sequences of the following two kinds of symbols (and parentheses and commas for ease of reading):

(1) A finite set F of *function symbols*, and

(2) A denumerable set V of *variables*.

We assume the reader is familiar with the concepts of *terms, ground terms, occurrences, subterms, substitutions, unifiers,* and *most general unifiers*. In what follows, we will denote the set of all terms constructed from F and V by $T(F,V)$, and the set of all the ground terms constructed from F by $T(F)$. The notation $t[s]$ represents a term with s as its subterm. In this context, $[s]$ represents a certain occurrence of s in $t[s]$. Thus, $t[s']$ denotes the term obtained by replacing the occurrence of s in $t[s]$ with s'. Similarly, we will use the notation $t[s_1,\ldots,s_n]$ to represent a term with $s_1,\ldots,s_n$ subterms, and $t[s'_1,\ldots,s'_n]$ for the term obtained by replacing each s_i in $t[s_1,\ldots,s_n]$ with s'_i. Substitutions are denoted by the greek letter θ, possibly with subscripts and primes.

Definition 2.1

A *term rewriting system* (*TRS*) is a finite set of pairs $l \to r$ of terms. An element $l \to r$ of a TRS is called a *rewrite rule*.

Definition 2.2

Let R be a TRS. A term t is said to be *reduced* to another term u with respect to R, if there exist a rewrite rule $l \to r$ and a substitution θ such that $c[\theta(l)] = t$ and $c[\theta(r)] = u$, denoted by $t \Rightarrow u$. We denote the reflexive transitive closure of $\Rightarrow$ by $\overset{*}{\Rightarrow}$.

Definition 2.3

Let R be a TRS. Two terms u and v are said to be *convergent* (with respect to R) if there exists a term t such that $u \overset{*}{\Rightarrow} t$ and $v \overset{*}{\Rightarrow} t$. A TRS is said to be *confluent* if t_1 and t_2 are convergent for any t and for any two reductions $t \overset{*}{\Rightarrow} t_1$ and $t \overset{*}{\Rightarrow} t_2$.

Definition 2.4

A TRS is said to *terminate* if there exists no infinite reduction $t_1 \Rightarrow t_2 \Rightarrow \cdots \Rightarrow t_n \Rightarrow \cdots$

Definition 2.5

A term t is said to be *irreducible* if there exists no term u such that $t \Rightarrow u$. An irreducible term s such that $t \overset{*}{\Rightarrow} s$ is called an *irreducible form* of t (with respect to R) and denoted by $t{\downarrow}$.

If R is a terminating TRS, then every term t has an irreducible form $t{\downarrow}$. Moreover, R is confluent if and only if the irreducible form $t{\downarrow}$ is unique. In this case, the TRS R is said to be *complete* and the irreducible form $t{\downarrow}$ is called the *normal form* of t.

Intuitively, a reduction step represents a computation step. Therefore, termination of a TRS means that every computation process finally stops and a certain result (i.e. an irreducible form) is obtained, while confluence of a TRS means that the result is unique. For this reason, completeness plays an important role in the study of TRSs (viewed as computation mechanisms) and the normal form of a term is sometimes called the value of the term.

Historically, however, the concept of TRS appeared as a decision procedure of word problems of universal algebra, where the completeness is very significant as well, because the decidability of the word problems depend on completeness of the TRS obtained by converting equational axioms to rewrite rules.

Definition 2.6

An *equational theory* is a set of pairs $t_1 \simeq t_2$ of terms satisfying the following conditions. (We use the symbol $\simeq$ for this purpose, and the symbol $=$ is taken to mean syntactical identity in this paper.)

(1) $t \simeq t$ for all terms t.

(2) If $t_1 \simeq t_2$, then $t_2 \simeq t_1$.

(3) If $t_1 \simeq t_2$, $t_2 \simeq t_3$, then $t_1 \simeq t_3$.

(4) If $t_1 \simeq t_2$, then $\theta(t_1) \simeq \theta(t_2)$ for any substitution θ.

(5) If $t_1 \simeq t_2$, then $s[t_1] \simeq s[t_2]$.

Any set E of pairs $l \simeq r$ of terms can be extended to an equational theory by considering the closure $T(E)$ of E with respect to the above conditions (1)-(5). In other words, the equational theory $T(E)$ is the least congruence including E. The set E is called an *(equational) axiom system* of the equational theory $T(E)$ and an element of E is called an *axiom*.

The word problem in an equational theory T involves the determination of whether $t_1 \simeq t_2$ for two arbitrary terms t_1 and t_2. Given an equational theory T, suppose that there exists a complete TRS such that $t_1 \simeq t_2$ if and only if $t_1{\downarrow} = t_2{\downarrow}$ for any two terms t_1 and t_2. Obviously, such a TRS can be viewed as an algorithm to solve the word problem of T. Knuth and Bendix devised a mechanical procedure to convert an

axiom system E to a complete TRS which solves the word problems of $T(E)$ [Knuth 70, Huet 81].

Before introducing the procedure, let us define critical pairs.

Definition 2.7

Let $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$ be rewriting rules and s be a non-variable subterm of l_2 such that l_1 and s have a most general unifier θ. Let $l_2 = c[s]$. The term $\theta(l_2)$ is called the *superposition* of l_1 on s in l_2. The pair $\theta(c[r_1]) \simeq \theta(r_2)$ is called a *critical pair* between $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$.

We are now ready to introduce the Knuth-Bendix completion procedure.

Procedure 2.8 Knuth-Bendix completion

Step 0: Set E to be the initially given axiom system. Set R to be empty. Go to Step 1.

Step 1: If E is empty, the current value of R is the desired TRS. Otherwise, go to Step 2.

Step 2: Remove a pair $t \simeq u$ from E. If the rule $t \rightarrow u$ or $u \rightarrow t$ can be added to R without violating termination, acquire it as a new rule and go to Step 3. Otherwise, stop; the procedure is unsuccessful.

Step 3: Remove all the rewrite rules $l \rightarrow r$ from R such that either l or r is reducible by the acquired new rule and append $l \simeq r$ to E instead. Go to Step 4.

Step 4: Append the acquired rule to R. Construct all the critical pairs between the acquired rule and all the rules in R (including the acquired rule itself) and append them to E. For each equation $t \simeq u$ in E, find irreducible forms $t\downarrow$ and $u\downarrow$ with respect to R, and set $\{\ t\downarrow \simeq u\downarrow \mid t\downarrow \neq u\downarrow, t \simeq u \in E\ \}$ to be the new E. Go to Step 1.

If the procedure terminates successfully, the resulting R is a complete TRS to solve the word problem of $T(E)$ for the initially given E.

3. Term rewriting system generator Metis

Metis is a TRS generator based on the completion procedure described in the previous section. It has a lot of functions required before, during, and after generation of TRSs for a very user-friendly system. In this section, we will describe several characteristic features of Metis.

3.1 Well-founded ordering of terms

As can be seen from the above description, a key point of the completion procedure is ensuring termination of a TRS. The standard way to assure termination of a system is to introduce a well-founded order on the objects of the system and show that the operations in the system always reduce the objects with respect to the order.

Well-founded orders $\prec$ on $T(F,V)$ with the following properties are usually used on TRSs.

(1) If $t_1 \prec t_2$, then $\theta(t_1) \prec \theta(t_2)$ for any substitution θ.

(2) If $t_1 \prec t_2$, then $s[t_1] \prec s[t_2]$.

Property (1) is called *stability* and (2) *monotonicity*. If there is a monotonic and stable well-founded order on $T(F,V)$ such that $l \succ r$ for every rule $l \rightarrow r$, it is obvious that the TRS terminates. There is a lot of research for such ordering methods, such as well-known Dershowitz's recursive path ordering [Dershowitz 82]. The original version of the recursive path ordering is defined on the set $T(F)$ of ground terms. Here, however, we extend the definition on the set $T(F,V)$ of all the terms.

Definition 3.1 Recursive path ordering

Let $<$ be a partial order on the set of function symbols F. The recursive path ordering $\prec$ of $T(F,V)$ is then defined recursively as follows:

(1) For a variable v, there are no terms t such that $t \prec v$.

(2) For a non-variable term $t = g(t_1, \cdots, t_n)$ and a term s, $s \prec t$ if and only if

(2-1) there exists j such that $s \preceq t_j$ or

(2-2) $s = f(s_1, \cdots, s_m)$ and $s_i \prec t$ for all i and

(2-2-1) $f < g$ or

(2-2-2) $f = g$ and $(s_1, \cdots, s_m) \overset{*}{\prec} (t_1, \cdots, t_n)$, where $\overset{*}{\prec}$ is the multi-set ordering [Dershowitz 79] induced by $\prec$.

In (2-2-2) of the above definition, employment of the multi-set ordering is not always necessary. If the function symbols f is varyadic (i.e. takes an arbitrary number of arguments) and the order of the arguments does not affect the value of the function (for example, $\sum$ and $\prod$ representing finite sum and product), the multi-set ordering is probably the most reasonable. However, if the function symbol f has a fixed arity, the lexicographic ordering is more suitable in many cases. There may be cases where the kachinuki ordering [Sakai 85] is the most appropriate.

Metis can handle any of these three versions of the recursive path ordering, namely multi-set, lexicographic, and kachinuki. The user can employ arbitrary combinations of them, function by function. As long as the lexicographic order is applied only to function symbols of fixed arity, any combination defines a monotone and stable well-founded order on $\mathcal{T}(F, V)$. Moreover, if the underlying order $<$ on F is total and the lexicographic or the kachinuki ordering are employed for any function symbol, then it is a total ordering on the limited domain $\mathcal{T}(F)$ of the ground terms, a very important property as we shall see later.

Metis converts axioms to rewrite rules $l \rightarrow r$ such that $l \succ r$. Metis allows the user to define the underlying partial order $<$ on F incrementally during the completion procedure. If the user knows little about the above ordering method, Metis can suggest what ordering is needed on F in order to orient an equation to a certain direction. Thus, when both are possible, the user just has to decide which direction an equation should be oriented to.

3.2 Associative and commutative operators

The weakest point of the Knuth-Bendix completion procedure is revealed by equations that cannot be converted to rules without violating the termination of the TRS. The most typical example of such axioms is the commutative laws, such as $A + B \simeq B + A$. Encounter with such an equation causes unsuccessful stop in Step 2 of the procedure. Metis has several countermeasures to deal with this situation. The general measures will be described later.

It is clearly the commutativity of operators that is the main source of the above failure. In many cases, commutative operators are also associative. Metis has a specific countermeasure effective only against the commutative laws combined with the associative laws of the same operators. A function symbol is called an *AC-operator* if it satisfies the associative and the commutative law. Metis is equipped with an algorithm of special unification for AC-operators (called AC-unification) devised by Fages [Fages 84] and can execute the AC-completion procedure based on Peterson and Stickel's principle [Peterson 81].

For example, if Metis is told that $+$ is an AC-operator, then the axioms $A + B \simeq B + A$ and $(A + B) + C \simeq A + (B + C)$ are acquired implicitly and AC-unification and AC-reduction are activated for $+$. Thus, Metis can generate $0 + Y + (-(X + Y)) \simeq (-X) + 0$ as a critical pair between the same two rules $(-X) + X \rightarrow 0$ by AC-unification, since

$$(-X) + X + Y + (-(X + Y)) \Rightarrow (-X) + 0$$

and

$$(-X) + X + Y + (-(X + Y)) \Rightarrow 0 + Y + (-(X + Y)).$$

If it has the rule $0 + A \rightarrow A$, the above critical pair is reduced to $Y + (-(X + Y)) \simeq -X$ by AC-reduction.

As shown in the above example, an AC-operator is supposed to be a binary function symbol and Metis allows us to use infix notation for binary function symbols. Inside Metis an AC-operator is treated as if it were varyadic. For example, the term $t_1 + \cdots + t_n$ is converted to $+(t_1, \ldots, t_2)$ with a varyadic function symbol $+$, in whatever order the operator $+$ is applied to the arguments. The multi-set ordering is assumed to be the ordering method for AC-operators unless otherwise specified, since the above treatment makes it the most reasonable ordering as mentioned in the previous section.

3.3 Orientation-free rules and S-strategy

There exist many equations other than commutative laws which cannot be converted to terminating rules. The approach of incorporating special unification algorithms for such equations has been studied systematically by Jouannaud and Kirchner [Jouannaud 84].

A simple trick to handle non-orientable equations is introducing a new function symbol. For example, if

the equation $A^2 \simeq A \times A$ cannot be oriented to either direction, a new function symbol *square* is introduced and the problematic equation is divided to the two equations $A^2 \simeq square(A)$ and $A \times A \simeq square(A)$. Thus, Metis can continue the completion procedure, since both equations can be oriented left to right. This technique seems to be too simple, but the effect is worth implementation [Knuth 70, Sakai 84].

A more radical remedy for such equations is adoption of orientation-free rules. This remedy is called the unfailing completion procedure [Hsiang 85, Bachmair 86]. Metis is equipped with an extended version of the unfailing completion procedure called S-strategy devised by Hsiang and Rusinowitch [Hsiang 85]. The S-strategy has enabled Metis to manipulate not only non-orientable equations, but also inequational axioms as well as equational axioms.

The S-strategy can be viewed as a kind of refutational theorem proving technique for systems of equations and inequations. Before introducing the S-strategy, we will extend the concepts of reduction and critical pairs and introduce the concept of extended narrowing and subsumption. Let us fix a monotonic and stable well-founded order $\prec$ on $T(F,V)$.

Definition 3.2

A term t is said to be *reduced* to another term u by an equation $l \simeq r$ (or $r \simeq l$), if $t \succ u$ and there exists a substitution θ such that $c[\theta(l)] = t$ and $c[\theta(r)] = u$. This reduction is called *extended reduction (by an equation)* and denoted also by $t \Rightarrow u$.

Definition 3.3

Let $l_1 \simeq r_1$ (or $r_1 \simeq l_1$) and $l_2 \simeq r_2$ (or $r_2 \simeq l_2$) be equations Let s be a non-variable subterm of l_2 such that l_1 and s have a most general unifier θ. Let $l_2 = c[s]$. If $\theta(l_1) \not\preceq \theta(r_1)$ and $\theta(l_2) \not\preceq \theta(r_2)$, then the pair $\theta(c[r_1]) \simeq \theta(r_2)$ is called an *extended critical pair* between $l_1 \simeq r_1$ (or $r_1 \simeq l_1$) and $l_2 \simeq r_2$ (or $r_2 \simeq l_2$).

If every rule $l \rightarrow r$ has the property that $l \succ r$, the above definitions are natural extensions of the ordinary reduction by a rule and the ordinary critical pairs between rules. For example, if $l \succ r$, the condition that $t \succ u$ in reducing t to u weakens the rewrite power of the equation $l \simeq r$ exactly to the same level as that of the rule $l \rightarrow r$, since $\prec$ is stable and monotonic. Similarly, if $l_1 \succ r_1$ and $l_2 \succ r_2$, the set of all extended critical pairs between equations $l_1 \simeq r_1$ and $l_2 \simeq r_2$ is equal to the set of all critical pairs between rules $l_1 \rightarrow r_1$ and $l_2 \rightarrow r_2$.

Definition 3.4

Let $l_1 \simeq r_1$ (or $r_1 \simeq l_1$) be an equation and $l_2 \not\simeq r_2$ (or $r_2 \not\simeq l_2$) be an inequation. Let s be a non-variable subterm of l_2 such that l_1 and s have a most general unifier θ. Let $l_2 = c[s]$. If $\theta(l_1) \not\preceq \theta(r_1)$, then the inequation $\theta(c[r_1]) \not\simeq \theta(r_2)$ is said to be *narrowed* from $l_2 \not\simeq r_2$ (or $r_2 \not\simeq l_2$) using $l_1 \simeq r_1$ (or $r_1 \simeq l_1$).

Definition 3.5

An equation $t \simeq u$ is said to be *subsumed* by other equations $l_1 \simeq r_1$ (or $r_1 \simeq l_1$), ..., $l_n \simeq r_n$ (or $r_n \simeq l_n$), if there exists a substitution θ such that

$$c[\theta(l_1), \ldots, \theta(l_n)] = t \text{ and } c[\theta(r_1), \ldots, \theta(r_n)] = u.$$

An inequation $t \not\simeq u$ is said to be *subsumed* by another inequation $l \not\simeq r$ (or $r \not\simeq l$), if there exists a substitution θ such that $\theta(l) = t$ and $\theta(r) = u$.

Unfailing completion is a modified version of ordinary completion employing extended critical pairs and extended reduction instead of the ordinary ones; and the S-strategy can be viewed as the unfailing completion with refutation by extended narrowing.

Procedure 3.6 S-strategy

Suppose that a system of equational and inequational axioms is given together with an equation or inequation to be solved (called the target formula).

Step 0: Set E to be the given axiom system plus the negation of the target formula (Skolemized if necessary). Set R to be empty. Go to Step 1.

Step 1: If E is empty, the current value of R is a complete set of equations and inequations deduced from the axioms and the negation of the target formula, in the sense that neither new equations nor new inequations can be derived. Since R is also consistent, the target formula cannot be deduced from the axioms. If E is not empty, go to Step 2.

Step 2: Remove an equation $t \simeq u$ or inequation $t \not\simeq u$ (called the ruling formula) from E. Go to Step 3.

Step 3: If the ruling formula is an equation, move all the equations $l \simeq r$ and all the inequations $l \not\simeq r$ from R to E such that either l or r is reducible by the ruling formula and remove

all the equations subsumed by the ruling formula from R. If the ruling formula is an inequation, remove all the inequations subsumed by the ruling formula from R. Go to Step 4.

Step 4: Append the ruling formula to R. Construct all the extended critical pairs and all the narrowed inequations between the ruling formula and all the equations and inequations in R. Append them to E. For each equation $t \simeq u$ or inequation $t \not\simeq u$ in E, find irreducible forms $t\downarrow$ and $u\downarrow$ with respect to equations in R. If there is an inequation $t \not\simeq u$ such that $t\downarrow$ and $u\downarrow$ are unifiable, then stop. A contradiction is detected and, therefore, the target formula is deduced from the originally given axiom system. Otherwise, let the new E be the set of equations $t\downarrow \simeq u\downarrow$ such that $t\downarrow \neq u\downarrow$ not subsumed by any equation in R and inequations $t\downarrow \not\simeq u\downarrow$ not subsumed by any inequation in R. Go to Step 1.

The unfailing completion differs from the S-strategy only in that it does not treat non-ground inequations. If the ordering $\prec$ is total on the set $T(F)$ of all the ground terms, the S-strategy is logically complete and, therefore, so is the unfailing completion.

4. Experiments

Let us begin with purely algebraic examples. The first example is the word problem of ring theory.

Example 4.1

Metis was given an AC-operator $+$ and a binary operator $*$, (not AC in general) with the following axioms:

(1) $0 + A = A$

(2) $(-A) + A = 0$

(3) $(A * B) * C = A * (B * C)$

(4) $(A + B) * C = A * C + B * C$

(5) $A * (B + C) = A * B + A * C$

We had Metis run the completion procedure in automatic mode. Metis obtained $(A * B) * C = A * (B * C)$ and $0 + A = A$ as the first and the second ruling formulas and converted them to the rules $(A * B) * C \to A * (B * C)$ and $0 + A \to A$, respectively. The third ruling formula $(-A) + A = 0$ could be oriented left to right by the recursive path ordering, if $0 < +$ or $0 < -$. So Metis asked the user which should be introduced.

```
[METIS] -> k
Knuth - Bendix  (automatic execution)
    r1:    (A*B)*C  ->  A*(B*C)
    r2:    0+A  ->  A
You can orient -A+A  ->  0 by:
  [1]   0 << +
  [2]   0 << -
  else  exit
```

After selecting $0 < +$, we had Metis continue the procedure.

```
select no ?  1
[ 0 << + is asserted.  ]
    r3:    -A+A  ->  0
    r4:    -(-A)  ->  A
    r5:    -(0)  ->  0
Which do you want to orient ?
  [1]   A*(B+C)  ->  A*B+A*C
  [2]   A*B+A*C  ->  A*(B+C)
  else  exit
```

The sixth ruling formula was the left distributive law and it could be oriented to either direction depending on the orderings on function symbols. Since we instructed Metis to convert it to the rule $A * (B + C) \to A * B + A * C$, the system automatically introduced $+ < -$ as the ordering on function symbols.

```
select no ?  1
[ + << * is asserted.  ]
    r6:    A*(B+C)  ->  A*B+A*C
    r7:    (A+B)*C  ->  A*C+B*C
    r8:    A+ -(B+A)  ->  -B
[ + << - is asserted.  ]
    r9:    -(A+(-B))  ->  B+(-A)
```

The nineth ruling formula can be converted to the rule $-(A + (-B)) \to B + (-A)$ if and only if $+ < -$. So Metis introduced the ordering without interaction.

```
    r10:  -(A+B)  ->  -A+(-B)
DELETE          r8
DELETE          r8*
DELETE          r9
    r11:  A*0+A*B  ->  A*B
    r12:  A*0  ->  0
DELETE          r11
```

```
DELETE          r11*
    r13:  0*A+B*A  ->  B*A
    r14:  0*A  ->  0
DELETE          r13
DELETE          r13*
    r15:  (-A)*B+A*B  ->  0
Which do you want to orient ?
  [1]   (-A)*B  ->  -A*B
  [2]   -A*B  ->  (-A)*B
  else  exit
select no ?  1
[ - << * is asserted.  ]
    r16:  (-A)*B  ->  -A*B
DELETE          r15
DELETE          r15*
    r17:  A*(-B)+A*B  ->  0
    r18:  A*(-B)  ->  -A*B
DELETE          r17
DELETE          r17*

   Knuth - Bendix terminated.
   Your system is COMPLETE.
```

The procedure terminated successfully. Here is the resulting complete TRS for the word problem of rings.

```
[METIS] -> list
<< state listing >>

        "ring"

  operators:
    + / AC  ( multiset ordering )
    0 / 0
    - / 1
    * / 2   ( left lexicographic )

  orderings:
    0  <  "+"  <  *,-
    "0"  <  +,*,-
    +,0  <  "-"  <  *
    +,-,0  <  "*"

  equations:
    No equations.

  rules:
    r1:    (A*B)*C  ->  A*(B*C)
    r2:    0+A  ->  A
    r2*:  A+0+B  ->  A+B
    r3:    -A+A  ->  0
    r3*:  A+(-B)+B  ->  A+0
    r4:    -(-A)  ->  A
    r5:    -(0)  ->  0
    r6:    A*(B+C)  ->  A*B+A*C
    r7:    (A+B)*C  ->  A*C+B*C
    r10:  -(A+B)  ->  -A+(-B)
    r12:  A*0  ->  0
    r14:  0*A  ->  0
    r16:  (-A)*B  ->  -A*B
    r18:  A*(-B)  ->  -A*B
```

Huet and Hullot developed a method to prove inductive theorems without explicit induction [Huet 82] using a modified version of the Knuth-Bendix completion procedure. Their method is called inductionless induction and is effective for many theorems which usually require explicit induction.

In order to use the method, ground terms have to be classified into two categories, namely, *constructor terms* which are always irreducible and constructed only of special function symbols called constructors, and *non-constructor terms* which are always reducible and include a function symbol other than constructors. To prove an inductive theorem, we add the statement as an axiom and execute the completion procedure. The statement is an inductive theorem if the process succeeds to completion without yielding any rules to rewrite constructor terms.

Metis was given an ordinary definition of the append operation for two lists and two different definitions of the reverse operation of a list.

```
[METIS] -> list rule
<< state listing >>

     "--- append & reverse ---"

  rules:
    r1:    append([],A)  ->  A
    r2:    rev([],A)  ->  A
    r3:    reverse([])  ->  []
    r4:    append([A|B],C)
           ->  [A|append(B,C)]
    r5:    rev([A|B],C)
           ->  rev(B,[A|C])
    r6:    reverse([A|B])
           ->  append(reverse(B),[A])
```

If we define $[\_|\_]$ (*cons*) and [] (*nil*) as the constructors, then the above conditions are satisfied. We added an equation $rev(A, []) = reverse(A)$ and had Metis execute the completion procedure.

```
[METIS] -> kB iNTERACTIVE
Knuth - Bendix (interactive execution)

Current ruling formula CAN be oriented.
    e7:    reverse(A)  =(<>)=  rev(A,[])
Which do you want to orient ?
  [1]   reverse(A)  ->  rev(A,[])
  [2]   reverse(A)  <-  rev(A,[])
  else  exit
Which ?  1
[ rev << reverse is asserted.  ]

Current ruling formula is ORIENTED.
    r7:    reverse(A)  ->  rev(A,[])
DELETE          r3
DELETE          r6

Current ruling formula CAN be oriented.
    e8:    rev(A,[B])
           =(<>)=  append(rev(A,[]),[B])
Which do you want to orient ?
  [1]   rev(A,[B])
           ->  append(rev(A,[]),[B])
  [2]   rev(A,[B])
           <-  append(rev(A,[]),[B])
  else  exit
Which ?  2
[ rev << append is asserted.  ]

Current ruling formula is ORIENTED.
    r8:    append(rev(A,[]),[B])
           ->  rev(A,[B])

Current ruling formula is ORIENTED.
    r9:    append(rev(A,[B]),[C])
           ->  rev(A,[B,C])

Current ruling formula is ORIENTED.
    e10:   append(rev(A,[B,C]),[D])
           =>=  rev(A,[B,C,D])
```

Since the current and the former ruling formulas suggested that a new lemma

$$append(rev(A, B), C) = rev(A, append(B, C))$$

would be useful, we added it.

```
[METIS/KB] -> new lEMMA
<< introduce a new lemma >>
> append(rev(A,B),C) = rev(A,append(B,C)).

Current ruling formula is ORIENTED.
    r10:   append(rev(A,B),C)
           ->  rev(A,append(B,C))
DELETE          r8
DELETE          r9

   Knuth - Bendix is terminated.
   Your system is COMPLETE.
```

The completion terminated and, therefore, both the target statement and the lemma inserted on the way were proved to be inductive theorems.

Several examples were taken from the theory of λ-calculus and combinators [Hindley 86, Barendregt 84]. In the theory of combinators, the combinator $\mathbf{K} = \lambda XY.\ X$ and $\mathbf{S} = \lambda XYZ.\ X * Z * (Y * Z)$ (as usual we assume that symbols $*$ standing for application of functions are left associative) are called basic combinators because all the λ-terms without free variables can be constructed from $\mathbf{S}$ and $\mathbf{K}$ only.

Example 4.2

It is well-known that the identity $\mathbf{I} = \lambda X.\ X$ is represented by $\mathbf{S} * \mathbf{K} * \mathbf{K}$. Metis was given the two axioms $\mathbf{K} * X * Y = X$ and $\mathbf{S} * X * Y * Z = X * Z * (Y * Z)$ for $\mathbf{K}$ and $\mathbf{S}$ to derive the identity. The problem can be expressed as $\exists I.\forall X.\ I * X = X$. Metis converted its negation to Skolemized form $A * \$1(A) \neq \$1(A)$ ($\$1$ is the so-called Skolem function).

```
[METIS] -> proVE ssTRATEGY TERMINAL
<< prove formulas by S-strategy >>

Formula > some(I,all(X, I*X = X )).

Try to prove formula :
    A* $1(A)  =/=  $1(A)
Enter S-strategy...

Current ruling formula is INEQUATION.
    r1:    A* $1(A)  <-/->  $1(A)

Current ruling formula is ORIENTED.
    r2:    k*A*B  ->  A
```

```
Current ruling formula is INEQUATION.
    r3:    A  <-/->  $1(k*A)

Current ruling formula is NOT orientable.
    r4:    s*A*B*C  <->  A*C*(B*C)

Current ruling formula is ORIENTED.
    r5:    s*k*A*B  ->  B

e13:  $1(s*k*A) =/= $1(s*k*A) [r5/r1]
        is a contradiction.  Then PROVED.
```

The first ruling formula was the target formula $A * \$1(A) \neq \$1(A)$ and the second was the axiom for $\mathbf{K}$, which was oriented left to right. The third formula was an extended narrowing from the first using the second, since $A = \mathbf{K} * A * \$1(\mathbf{K} * A) \neq \$1(\mathbf{K} * A)$. The fourth was the axiom for $\mathbf{S}$ which could not be oriented. The fifth was an extended critical pair between the fourth and the second, since $\mathbf{S} * \mathbf{K} * A * B = \mathbf{K} * B * (A * B) = B$. Using this, a contradictory narrowing was obtained from the first ruling formula. By examining this process, we easily find all terms of the form $\mathbf{S} * \mathbf{K} * A$ are equal to the identity function, and $\mathbf{S} * \mathbf{K} * \mathbf{K}$ is merely an instance of such terms.

Example 4.3

Next, we had Metis try to prove the fixed-point theorem, i.e. that there exists a fixed-point for any combinator, with the existence of the combinators $\mathbf{B} = \lambda XYZ.\ X * (Y * Z)$ of composition of functions and $\mathbf{M} = \lambda X.\ X * X$ of self-application, which are defined by $\mathbf{B} = \mathbf{S} * (\mathbf{K} * \mathbf{S}) * \mathbf{K}$ and $\mathbf{M} = \mathbf{S} * \mathbf{I} * \mathbf{I}$. Metis was given the axioms $\mathbf{B} * X * Y * Z = X * (Y * Z)$ and $\mathbf{M} * X = X * X$. The theorem can be expressed as $\forall F. \exists P.\ F * P = P$.

```
[METIS] -> list all
<< state listing >>
  operators:
    * / 2 ( left lexicographic )
    b / 0
    m / 0

  orderings:
    No orderings

  equations:
    e1:    m*A  =  A*A      [axiom]
    e2:    b*A*B*C  =  A*(B*C)      [axiom]
  rules:
    No rules.

[METIS] -> prove sstrategy terminal
<< prove equations by S-strategy >>

Equation > all(F,some(P, F*P = P )).

Try to prove equation :
    $1*A  =/=  A
Enter S-strategy...

Current ruling formula is INEQUATION.
    r1:    $1*A  <-/->  A

Current ruling formula is NOT orientable.
    e1:    m*A  =  A*A
```

Since the above ruling formula could not be oriented, we let Metis introduce a new function symbol s and rewrite both $A * A$ and $\mathbf{M} * A$ to $s(A)$. Acquisition of the new function symbol and orientation of new equations was done interactively as follows:

```
[METIS/PROVE/S-STRA] -> new function
<< introduce a new function >>
Operator?s
[ e3:    m*A  =  s(A) is asserted.  ]
[ e4:    A*A  =  s(A) is asserted.  ]

Current ruling formula CAN be oriented.
    e4:    A*A  =(<>)=  s(A)
[METIS/PROVE/S-STRA] -> suggestion current
<< suggestion for ordering >>
Which do you want to orient ?
  [1]   A*A  ->  s(A)
  [2]   A*A  <-  s(A)
  else  exit
Which ?  1
[ s << * is asserted.  ]

Current ruling formula is ORIENTED.
    r2:    A*A  ->  s(A)

Current ruling formula is ORIENTED.
    r3:    m*A  ->  s(A)

Current ruling formula is INEQUATION.
    r4:    s($1)  <-/->  $1

Current ruling formula is ORIENTED.
```

```
    r5:     b*A*B*C  ->  A*(B*C)

Current ruling formula is ORIENTED.
    r6:     s(b)*A*B  ->  b*(A*B)

Current ruling formula is ORIENTED.
    r7:     s(b*A)*B  ->  A*(b*A*B)

Current ruling formula is ORIENTED.
    r8:     A*(B*(b*A*B))  ->  s(b*A*B)

Current ruling formula is ORIENTED.
    r9:     s(s(b))*A  ->  b*(s(b)*A)

Current ruling formula is INEQUATION.
    r10:  s(b*$1*A)  <-/->  A*(b*$1*A)

e32:  s(b*$1*m) =/= s(b*$1*m) [r3/r10]
        is a contradiction.  Then PROVED.
```

Metis finally found a contradictory inequation. The inequation was obtained by substituting $\mathbf{M}$ to A in r10 and rewriting the right hand side by r3. The inequation r10 was from r1 and r8, since

$$s(\mathbf{B}*\$1*A) = \$1*(A*(\mathbf{B}*\$1*A)) \neq A*(\mathbf{B}*\$1*A).$$

and the rule r8 was from r2 and r5, since

$$A*(B*(\mathbf{B}*A*B)) = \mathbf{B}*A*B*(\mathbf{B}*A*B) = s(\mathbf{B}*A*B).$$

Examining this process of refutation showed us that m*(B*$1*m) is the value substituted to the original variable A in the inequality obtained by the negation of the target formula. In fact, it is a fixed point of $1, since

$$\begin{aligned}\mathbf{M}*(\mathbf{B}*\$1*\mathbf{M}) &= \mathbf{B}*\$1*\mathbf{M}*(\mathbf{B}*\$1*\mathbf{M}) \\ &= \$1*(\mathbf{M}*(\mathbf{B}*\$1*\mathbf{M}))\end{aligned}$$

REFERENCES

[Bachmair 86] Bachmair, L., Dershowitz, N., and Plaisted, D.A.: *Completion without failure*, private communication (1986)

[Barendregt 84] Barendregt, H.P. *The lambda calculus:Its syntax and semantics*, Studies in Logic and the Foundations of Mathematics, vol. 103, North-Holland, revised edition (1984)

[Dershowitz 79] Dershowitz, N. and Manna, Z.: *Proving termination with multiset orderings*, C.ACM 22 (1979) 465-467

[Dershowitz 82] Dershowitz, N.: *Orderings for term-rewriting systems*, Theoretical Computer Science 17 (1982) 279-301

[Fages 84] Fages, F.: *Associative-commutative unification*, 7th. International Conference on Automatic Deduction, LNCS 170 (1984) 194-208

[Hindley 86] Hindley, J.R. and Seldin, J.P.: *Introduction to combinators and λ-calculus*, London Mathematical Society Student Text 1, Cambridge University Press (1986)

[Hsiang 85] Hsiang, J. and Rusinowitch, M.: *On word problems in equational theories*, private communication (1985)

[Huet 81] Huet, G.: *A complete proof of correctness of the Knuth-Bendix completion algorithm*, J. Computer and System Science 23 (1981) 11-21

[Huet 82] Huet, G. and Hullot, J-M.: *Proofs by induction in equational theories with constructors*, J. Computer and System Science, 25 (1982) 239-266

[Jouannaud 84] Jouannaud, J-P. and Kirchner, H.: *Completion of a set of rules modulo a set of equations*, 11th ACM POPL, 1984

[Knuth 70] Knuth, D. E., Bendix, P. B.: *Simple word problems in universal algebras*, Computational problems in abstract algebra, J. Leech (ed), Pergamon Press, Oxford, (1970) 263-297. also in: Automated Reasoning 2 (Siekmann and Wrightson eds.), Springer (1983)

[Peterson 81] Peterson, G.E. and Stickel, M.: *Complete sets of reductions for equational theories with complete unification algorithms*, J.ACM, vol 28 (1981) 233-264

[Sakai 84] Sakai, K.: *An ordering method for term rewriting systems*, ICOT TR-062 (1984)

[Sakai 85] Sakai, K.: *Knuth-Bendix algorithm for Thue system based on kachinuki ordering*, ICOT TM-0087 (1985)