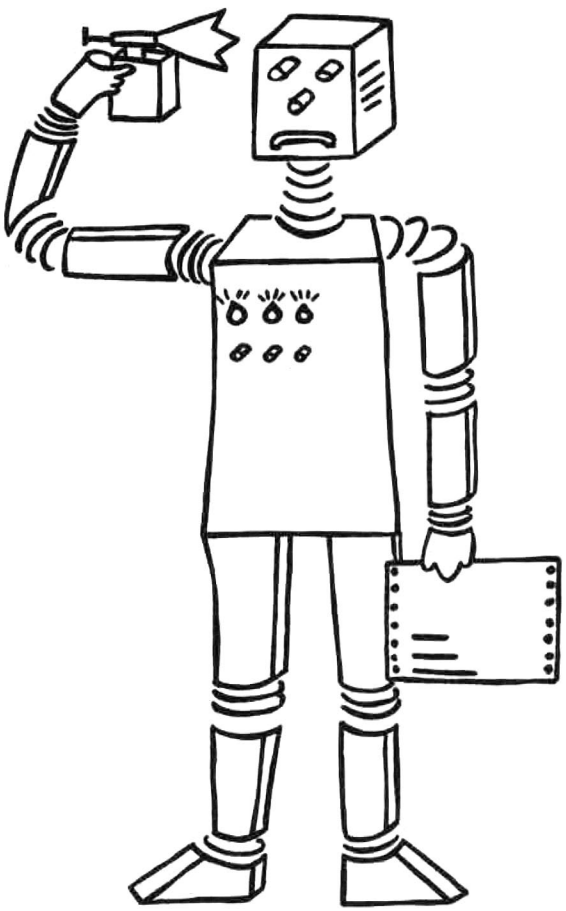
Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern 1, W. Germany

SEKI-REPORT

Artificial
Intelligence
Laboratories



Completion of Globally Finite Term Rewriting Systems for Inductive Proofs

Richard Göbel

Juni 1986 SEKI-REPORT SR-86-06

Completion of Globally Finite Term Rewriting Systems for Inductive Proofs

Richard Göbel

/85 Mai 1985

Abstract

The Knuth-Bendix Algorithm (KBA) is not able to complete term rewriting systems with cyclic rules such as the commutativity. This kind of rules cause cycles in a reduction chain. This problem may be solved by an extension of the KBA for globally finite term rewriting systems. For a globally finite term rewriting system, cycles may occur in a reduction chain, but for each term there is only a finite set of reductions. A confluent and globally finite term rewriting sytem R provides a decision procedure for equality induced by R:

Two terms are equal iff there is a common term in their reduction sets.

This extension requires new methods for testing the global finiteness and a new confluence test, because local confluence does not imply global confluence for globally finite relations. In this paper we give a theoretical framework for this extension. We will show how this theory can be applied to term rewriting systems, if we are mainly interested in the initial algebra which is induced by the set of rules.

Notation and Basic Definitions

We assume familiarity of the reader with the basic proofs and results of the Knuth-Bendix Algorithm (e.g. [HU 77], [HO 80], [KB 70]).
We denote by:

V	set of variables
$F = C \cup D$	set of function symbols, which may be splitted into constructors C and defined functions D
t	terms constructed by symbols from V and F
T	set of all terms constructed by symbols from V and F
GT	set of all ground terms constructed by symbols from F
CT	set of all constructor terms constructed by symbols from V and C
CGT	set of all constructor ground terms constructed by symbols from C
u, v, w	occurences in terms
$O(t)$	set of all occurences of t
$V(t)$	set of all variables of t
$\alpha \longrightarrow \beta,\ \gamma \longrightarrow \delta$	rewrite rules
σ	substitutions
Σ	set of all substitutions Σ
R	set of rules
$t \longrightarrow_R t'$	t is reducible in one step to t' by R
$t \xrightarrow{n}_R t'$	t is reducible in n steps to t' by R
$t \xrightarrow{*}_R t'$	t is reducible in a finite number of steps to t' by R or $t = t'$
$\vdash\!\dashv_R$	symmetric closure of $\longrightarrow_R$
$\vdash\!\dashv^*_R$	transitive closure of $\vdash\!\dashv_R$
$\vdash\!\dashv^*_{R,G}$	$\vdash\!\dashv^*_R$ restricted to ground terms

Definitions

t is **linear**, iff: $\forall\, x \in V(t) : \forall\, u, v \in O(t) : t/u = x \wedge t/v = x \Longrightarrow u = v$

A variable x is linear in a term t, iff: $\forall\, u, v \in O(t) : t/u = x \wedge t/v = x \Longrightarrow u = v$

A rule is linear, iff the left hand side of the rule is linear.

R is **terminating**, iff for any term t, there is no infinite reduction chain derivable from t.

R is **globally finite**, iff for any term t, the reduction set of t is finite.

R is **confluent**, iff: $\forall\, t, t_1, t_2 : t \xrightarrow{*}_R t_1 \wedge t \xrightarrow{*}_R t_2 \Longrightarrow \exists\, t' : t_1 \xrightarrow{*}_R t' \wedge t_2 \xrightarrow{*}_R t'$

R is **locally confluent**, iff: $\forall\, t, t_1, t_2 : t \longrightarrow_R t_1 \wedge t \longrightarrow_R t_2 \Longrightarrow \exists\, t' : t_1 \xrightarrow{*}_R t' \wedge t_2 \xrightarrow{*}_R t'$

A term t is in **R-normal form**, iff: $\forall\, t' : t \xrightarrow{*}_R t' \Longrightarrow t = t'$

A term t is in **R-normal form modulo cycles**, iff: $\forall\, t' : t \xrightarrow{*}_R t' \Longrightarrow t' \xrightarrow{*}_R t$

For this paper we assume further, that no left hand side of a rule consists of a single variable.

(1) Introduction

During the completion of term rewriting systems, the Knuth-Bendix Algorithm creates sometimes critical pairs which can not be directed to terminating rules (cyclic rules). For example, if we complete the following set of rules, the commutativity axiom is created as a critical pair:

(R1) $f(f(x,y),z) \longrightarrow f(x,f(y,z))$
(R2) $f(0,x) \longrightarrow x$
(R3) $f(x,0) \longrightarrow x$
(R4) $f(x,x) \longrightarrow 0$

After a few completion steps the Knuth-Bendix Algorithm (KBA) creates the commutativity from this set of rules and stops with failure.
If we want to complete systems with cyclic rules, we have to extend the Knuth-Bendix Algorithm. There are two ways for this extension:

(i) Separate the cyclic rules from the term rewriting system and perform reduction steps on equivalence classes which are created by these cyclic rules. This extension preserves the finite termination property but in general requires a unification algorithm for the equational theory induced by the cyclic rules (e.g. [JK 84]).

(ii) Drop the finite termination property and extend the KBA for globally finite term rewriting systems. This extension requires methods for proving the global finiteness of term rewriting systems and for proving the confluence of a globally finite term rewriting system.

Here we will discuss how the global finiteness of a term rewriting system can be proved by weak term orderings. Then we will give confluence properties for arbitrary globally finite relations and apply these results to term rewriting systems for proofs in the initial algebra.

(2) Proving the global finiteness for term rewriting systems

A globally finite term rewriting system R defines a preordering on the set of terms $\geq$ where the set $\{ t' \mid t \geq t' \}$ is finite for all terms t :

$$t \geq t' \iff : t \xrightarrow{*}_R t'$$

Therefore, we may prove the global finiteness of a term rewriting system R by using such a preordering $\geq$:

$$(\forall\, t, t' : t \longrightarrow_R t' \Rightarrow t \geq t') \Rightarrow R \text{ globally finite}$$

For a test which bases on comparing left hand sides and right hand sides of rules, the ordering has to satisfy more properties:

(2.1) Definition
Let $\geq$ be a preordering on terms ($\geq$ is reflexive and transitive). Then $\geq$ is a weak term ordering, iff:

- $\forall\, t : \exists\, n : |\{ t' \mid t \geq t' \}| < n$
- $\forall\, t, t_1, t_2 : t_1 \geq t_2 \Rightarrow t[u \leftarrow t_1] \geq t[u \leftarrow t_2]$
- $\forall\, t, t' : \forall\, \sigma : t \geq t' \Rightarrow \sigma(t) \geq \sigma(t')$

(2.2) Theorem
Let R be a term rewriting system and $\geq$ a weak term ordering. Then R is globally finite, if for every rule $\alpha \longrightarrow \beta$ from R $\alpha \geq \beta$ holds.

Proof
first we will prove: $t \longrightarrow_R t' \Rightarrow t \geq t'$

$$t \longrightarrow_R t' \Rightarrow \exists\, \alpha \longrightarrow \beta \in R : t / u = \sigma(\alpha) \wedge t' = t[u \leftarrow \sigma(\beta)]$$

with $\alpha \geq \beta$:
$\Rightarrow \sigma(\alpha) \geq \sigma(\beta)$
$\Rightarrow t[u \leftarrow \sigma(\alpha)] \geq t[u \leftarrow \sigma(\beta)]$
$\Rightarrow t \geq t'$

with the transitivity of $\geq$ we get :

$$t \xrightarrow{*}_R t' \Rightarrow t \geq t'$$

and because $\{ t' \mid t \geq t' \}$ is finite for any term t then $\{ t' \mid t \xrightarrow{*}_R t' \}$ is also finite

This kind of orderings can be created by combinig a classical term ordering $>$ and a congruence $\sim$ on terms, if the relation $\sim$ is compatible with the ordering $>$. The next defintion gives the classical axioms for a term ordering:

(2.3) Definition

Let $>$ be a strict ordering on terms ($>$ is irreflexive, asymmetric and transitive) with:

(1) $>$ is well founded

(2) $\forall\, t, t_1, t_2 : t_1 \geq t_2 \Rightarrow t[u \leftarrow t_1] \geq t[u \leftarrow t_2]$

(3) $\forall\, t, t' : \forall\, \sigma : t > t' \Rightarrow \sigma(t) > \sigma(t')$

Then $>$ is a term ordering.

(2.4) Lemma

Let $>$ be a term ordering and $\sim$ a congruence on terms with:

(1) $\forall\, t_1, t_2, t_3 : (t_1 \sim t_2 \wedge t_2 > t_3) \Rightarrow t_1 > t_3$

(2) $\forall\, t_1, t_2, t_3 : (t_1 > t_2 \wedge t_2 \sim t_3) \Rightarrow t_1 > t_3$

(3) $\forall\, t_1, t_2 : \forall\, \sigma : t_1 \sim t_2 \Rightarrow \sigma(t_1) \sim \sigma(t_2)$

(4) $\forall\, t : \exists\, n : |\{ t' \mid t \sim t' \}| < n$

and we define the relation $\geq$ as :

$t_1 \geq t_2 \Leftrightarrow : t_1 \sim t_2 \vee t_1 > t_2$

Then $\geq$ is a weak term ordering.

Proof

- $\geq$ reflexive

  $t \sim t \Rightarrow t \geq t$ ($\sim$ congruence)

- $\geq$ transitive

  $t_1 \geq t_2 \wedge t_2 \geq t_3$

  (i) $t_1 \sim t_2 \wedge t_2 \sim t_3 \Rightarrow t_1 \sim t_3 \Rightarrow t_1 \geq t_3$ ($\sim$ congruence)

  (ii) $t_1 \sim t_2 \wedge t_2 > t_3 \Rightarrow t_1 > t_3 \Rightarrow t_1 \geq t_3$ (1)

  (iii) $t_1 > t_2 \wedge t_2 \sim t_3 \Rightarrow t_1 > t_3 \Rightarrow t_1 \geq t_3$ (2)

  (iv) $t_1 > t_2 \wedge t_2 > t_3 \Rightarrow t_1 > t_3 \Rightarrow t_1 \geq t_3$ ($>$ transitive)

- $\forall\, t : \exists\, n : |\{ t' \mid t \geq t' \}| < n$

  $\{ t' \mid t \geq t' \} = \{ t' \mid t > t' \} \cup \{ t' \mid t \sim t' \}$ (definition of $\geq$)

  Since $\{ t' \mid t > t' \}$ and $\{ t' \mid t \sim t' \}$ are finite, then $\{ t' \mid t \geq t' \}$ is finite

- $\forall\, t, t' : t \geq t' \Rightarrow t[u \leftarrow t] \geq t[u \leftarrow t']$

  (i) $t > t' \Rightarrow t[u \leftarrow t] > t[u \leftarrow t'] \Rightarrow t[u \leftarrow t] \geq t[u \leftarrow t']$ ($>$ term ordering)

  (ii) $t \sim t' \Rightarrow t[u \leftarrow t] \sim t[u \leftarrow t'] \Rightarrow t[u \leftarrow t] \geq t[u \leftarrow t']$ ($\sim$ congruence)

- $\forall\, t, t' : \forall\, \sigma : t \geq t' \Rightarrow \sigma(t) \geq \sigma(t')$

  (i) $t > t' \Rightarrow \sigma(t) > \sigma(t') \Rightarrow \sigma(t) \geq \sigma(t')$ ($>$ term ordering)

  (ii) $t \sim t' \Rightarrow \sigma(t) \sim \sigma(t') \Rightarrow \sigma(t) \geq \sigma(t')$ (3)

Here are two examples of orderings, that can be extented by a congruence:

- Length Ordering
  $t > t' \Longleftrightarrow$: in t' are less symbols than in t and every variable occures fewer times in t' than in t
  $t \sim t' \Longleftrightarrow$: in t' and t are the same number of symbols and every variable occurs in both terms at the same number of occurences

- Recursive Path Ordering
  $t > t' \Longleftrightarrow$: $t >_{RPO} t'$ as defined in [DE 82]
  $t \sim t' \Longleftrightarrow$: we get t' from t by the permutation of arguments

For the confluence test in chapter 4 we have to distinguish between cyclic and reduction rules. If we apply a reduction rule to a term t and get a term t' as the result, then t is no longer in the reduction set of t'. After applying a cyclic rule to a term t, t is still derivable from the result of the application. Therefore, a reduction rule decreases the size of a term in some term ordering, where a cyclic rule preserves the size of a term. The next lemma shows, how a globally finite term rewriting system can be splitted into cyclic and reduction rules by using a term ordering.

(2.5) Definition
Let R be a globally finite term rewriting system.
- A rule $\alpha \longrightarrow \beta$ from R is a reduction rule, iff:
  $\forall\, t, t' : \forall\, \sigma : t / u = \sigma(\alpha) \wedge t' = t[u \leftarrow \sigma(\beta)] \Longrightarrow t' \overset{+}{\rightarrow}_R t$
- A rule $\alpha \longrightarrow \beta$ from R is a cyclic rule, iff:
  $\forall\, t, t' : \forall\, \sigma : t / u = \sigma(\alpha) \wedge t' = t[u \leftarrow \sigma(\beta)] \Longrightarrow t' \overset{*}{\longrightarrow}_R t$

(2.6) Lemma
Let R be a globally finite term rewriting system and $\geq$ a weak term ordering with:
(1) $\forall\, \alpha \longrightarrow \beta \in R : \alpha \geq \beta$
(2) $(t > t' \Longleftrightarrow : t \geq t' \wedge \neg\, t' \geq t) \Longrightarrow >$ is a term ordering
Then:
- $\alpha \longrightarrow \beta \in R$ is a reduction rule, if $\alpha \geq \beta$ and $\neg\, \beta \geq \alpha$
- $\alpha \longrightarrow \beta \in R$ is a cyclic rule, iff $\beta \overset{*}{\longrightarrow}_R \alpha$

Note:
- condition (2) is satisfied, if $\geq$ is created by combining a term ordering and a congruence
- if these tests cannot be applied for a rule $\alpha \longrightarrow \beta \in R$, then we may force this rule to be cyclic by adding $\beta \longrightarrow \alpha$ to the rule set, if $\beta \geq \alpha$ holds.

Proof
(i) $\neg\, \beta \geq \alpha$ holds for $\alpha \longrightarrow \beta \in R$
Assume: $\exists\, t, t' : t \longrightarrow_{\{\alpha \longrightarrow \beta\}} t' \wedge t' \overset{*}{\longrightarrow}_R t$
$\Longrightarrow t' \geq t \wedge t \geq t'$
but with $\neg\, \beta \geq \alpha$ and (2) we get $t > t'$:
$\Longrightarrow \neg\, t' \geq t$ ↯

(ii) ($\Longrightarrow$)

Since we may derive β from α by the rule $\alpha \longrightarrow \beta$, this case is obvious

($\Longleftarrow$)

$t \longrightarrow_{\{\alpha \longrightarrow \beta\}} t' \Longrightarrow \exists\, u:\ t/u = \sigma(\alpha) \wedge t' = t[u \leftarrow \sigma(\beta)]$

with $\beta \xrightarrow{*}_R \alpha$:

$\Longrightarrow \sigma(\beta) \xrightarrow{*}_R \sigma(\alpha)$

$\Longrightarrow t[u \leftarrow \sigma(\beta)\,] \xrightarrow{*}_R t[u \leftarrow \sigma(\alpha)]$

$\Longrightarrow t' \xrightarrow{*}_R t$

(3) Confluence Properties of Globally Finite Relations

In this chapter we discuss confluence properties for arbitrary relations, which are not necessarily induced by term rewriting systems.

The local confluence test which does imply the confluence of a terminating relation, cannot be applied for globally finite term rewriting systems. Let us consider the following example:

This globally finite relation is locally confluent but not confluent.

Therefore, we need a stronger property which implies the confluence of globally finite relations. This stronger property has to prove, whether from all exits of a cycle, we can reach the same element.

(3.1) Definition

Let R be a globally finite relation. R is locally confluent modulo cycles iff:

$$\forall\, t_1, t_2, t'_1, t'_2 : t_1 \xrightarrow{*}_R t_2 \wedge t_2 \xrightarrow{*}_R t_1 \wedge t_1 \longrightarrow_R t'_1 \wedge t_2 \longrightarrow_R t'_2$$
$$\Longrightarrow \exists\, t' : t'_1 \xrightarrow{*}_R t' \wedge t'_2 \xrightarrow{*}_R t'$$

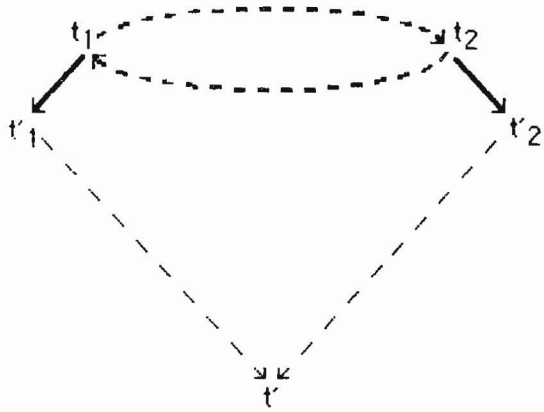


Fig. 3.1 Local Confluence Modulo Cycles

(3.2) Theorem

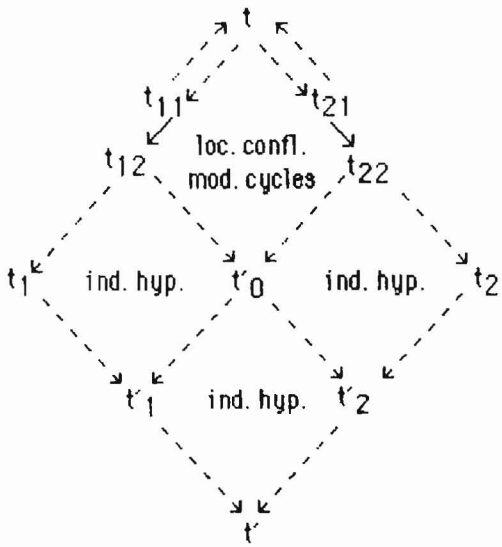
Let R be a globally finite relation. R is globally confluent, iff R is locally confluent modulo cycles.

Proof

($\Longrightarrow$) obvious

(⟸) This case can be proven by induction on the number of derivable elements from an element t.

Since the base case is obvious (no element is derivable from t), we will shortly sketch the induction step. Assume that t_1 and t_2 are derivable in an arbitrary number of steps from t. If t is in the reduction set of t_1 (t_2), then t_1 (t_2) can be reduced to t_2 (t_1). In the other case there are terms $t_{11}, t_{12}, t_{21}, t_{22}$ on the reduction chain to t_1 and t_2, where t is in the reduction set of t_{11} and t_{21}, but not in the reduction set of t_{12} and t_{22}, thereby t_{12}, t_{22} can be derived in one step from t_{11} and t_{21}. Then, less elements are derivable from t_{12} and t_{22} than from t, and we get:



Unfortunately, the local confluence modulo cycles cannot be proven by a simple test, because the cycles may be of arbitrary size. In general, the confluence of globally finite term rewriting systems is not decidable [NM 84]. Therefore, we have to find stronger properties, which are sufficient but not necessary for proving the confluence property.
In [JK 84] the coherence property is introduced for proving the confluence of equational term rewriting systems. The results of this paper can be carried over to globally finite term rewriting systems.
Let R be a globally finite and coherent relation. An exit element e of a cycle in R is an element where another element e', which is not member of this cycle, is derivable in one step from e. Then, all elements of a cycle with at least one exit element have to be also exit elements.
The next definition introduces two coherence properties:

(3.3) Definition
Let R be a globally finite relation.

- R is **coherent** iff:

$$\forall\, t, t_1, t_2 : t \longrightarrow_R t_1 \wedge t \xrightarrow{*}_R t_2 \wedge t_2 \xrightarrow{*}_R t' \wedge t_1 \not\xrightarrow{*}_R t$$
$$\Longrightarrow \exists\, t'_2 : t_2 \longrightarrow_R t'_2 \wedge t'_2 \not\xrightarrow{*}_R t_2 \wedge \exists\, t' : t'_2 \xrightarrow{*}_R t' \wedge t_1 \xrightarrow{*}_R t'$$

- R is **locally coherent** iff:

$$\forall\, t, t_1, t_2 : t \longrightarrow_R t_1 \wedge t \longrightarrow_R t_2 \wedge t_2 \xrightarrow{*}_R t \wedge t_1 \xrightarrow{*}_R t$$
$$\Longrightarrow \exists\, t'_2 : t_2 \longrightarrow_R t'_2 \wedge t'_2 \xrightarrow{*}_R t_2 \wedge \exists\, t' : t'_2 \xrightarrow{*}_R t' \wedge t_1 \xrightarrow{*}_R t'$$

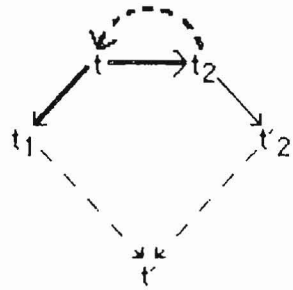


Fig. 3.2 Local Coherence

For confluent relations, these two properties are equivalent:

(3.4) Lemma
Let R be a globally finite and confluent relation. R is coherent iff it is locally coherent.

The proof of this lemma can be done by induction on the minimal number of steps between two elements from a cycle. The next diagram sketches the induction step for this proof:

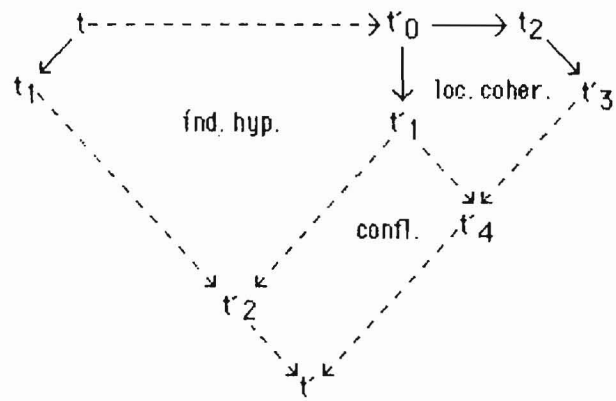


Fig. 3.3

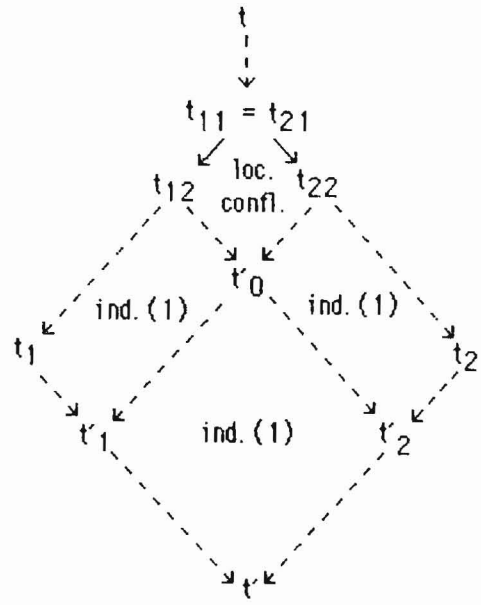
(3.5) Theorem
Let R be a globally finite relation. R is confluent, if R is locally coherent and locally confluent.

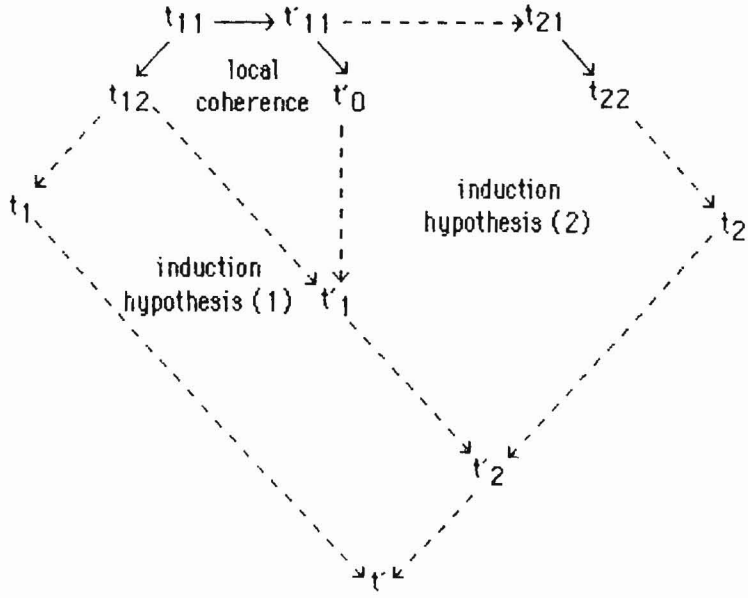
Proof

We prove this theorem by induction on the number of derivable elements from an element t (induction 1). The base case is obvious (no elements are derivable from t). For the induction step we assume, that t_1 and t_2 are derivable from t. If t is in the reduction set of t_1 or t_2 this completes the proof. Otherwise there are terms t_{11}, t_{12}, t_{21} and t_{22}, where t_{11}, t_{21} are in the same cycle as t, t_{12} and t_{22} are not in the cycle of t and t_{12}, t_{22} are derivable in one step from t_{11} and t_{21}. For the proof of this case, we need another induction on the mininal number n of derivation steps between t_{11} and t_{21}:

$$t_{11} \xrightarrow{n} t_{21} \wedge t_{21} \xrightarrow{*} t_{11} \wedge t_{11} \xrightarrow{*} t_1 \wedge t_{21} \xrightarrow{*} t_2 \Longrightarrow \exists\, t' : t_1 \xrightarrow{*} t' \wedge t_2 \xrightarrow{*} t'$$

The proof for the base case, where $t_{11} = t_{21}$, is given by the following diagram and the induction step by the next diagram:





A globally finite and confluent relation is not necessarily coherent, therefore we may find weaker properties for proving the confluence. A completion procedure with a weaker confluence test might create less rules than a completion procedure which requires the coherence property. On the other hand, a completion procedure needs an efficient algorithm for reducing terms. If we do not want to traverse all elements of a cycle during the reduction of an element, we need the coherence property for this relation.
In the next defintion, we split a globally finite relation R into a cyclic subrelation C(R) and a reducing subrelation R(R). Then we will prove, that for confluent and coherent relations R, it is sufficient to use the subrelation R(R) for reducing an element t.

(3.6) Definition
Let R be a globally finite relation. The reducing subrelation R(R) and the cyclic subrelation C(R) of R are defined as follows:

$$t \longrightarrow_{R(R)} t' \iff : t \longrightarrow_R t' \wedge t' \not\xrightarrow{*}_R t$$

$$t \longrightarrow_{C(R)} t' \iff : t \longrightarrow_R t' \wedge t' \xrightarrow{*}_R t$$

Note:
- $R = C(R) \cup R(R)$
- R(R) is terminating

Now we will prove, that it is sufficient to use R(R) instead of R for reducing terms.

(3.7) Lemma
Let R be a globally finite, coherent and confluent relation. An element t is in R(R) normal form, iff t is in R normal form modulo cycles.

Proof
($\Longrightarrow$) obvious
($\Longleftarrow$)
proof by contraposition:

t is not in R normal form modulo cycles

$\Longrightarrow \exists\, t_1 : t \xrightarrow{*}_R t_1 \wedge t_1 \xrightarrow{*}_R t \wedge \exists\, t_2 : t_1 \longrightarrow_R t_2 \wedge t_2 \xrightarrow{*}_R t_1$

(there is an element t_2 below this cycle)

$\Longrightarrow \exists\, t'_1 : t \longrightarrow_R t'_1 \wedge t'_1 \xrightarrow{*}_R t$ (R coherent)

$\Longrightarrow t \longrightarrow_{R(R)} t'_1$

$\Longrightarrow$ t is not in R(R) normal form

(3.8) Lemma
Let R be a globally finite, confluent and coherent relation.
Then:

$\forall\, t, t' :$ $t \xrightarrow{*}_R t' \wedge t'$ is in R normal form modulo cycles $\Longrightarrow t{\downarrow}R(R) \xrightarrow{*}_{C(R)} t'$

Proof
$t{\downarrow}R(R)$ is in R(R)-normal form
$\Longrightarrow t{\downarrow}R(R)$ is in R normal form modulo cycles
with t' is an R-normal form modulo cycles of t:
$\Longrightarrow t{\downarrow}R(R) \xrightarrow{*}_{C(R)} t'$ (R confluent)

(4) Confluence of Term Rewriting Systems

Now, we will apply the results of chapter (3) to term rewriting systems. Unfortunately, the coherence property seems to be too restrictive for arbitrary term rewriting systems. For example, if we apply a cyclic rule to a subterm which can be matched by a nonlinear variable of a reduction rule, the reduction rule cannot be applied to the result of the application of the reduction rule. The cyclic rule can be applied in arbitrary depth of the term, therefore the term rewriting system cannot be made coherent by adding a finite set of rules.

Example

(1) $f(x,I(x)) \longrightarrow 0$

(2) $f(x,y) \longrightarrow f(y,x)$

$$f(h(f(a,b)),I(h(f(a,b)))) \longrightarrow f(h(f(b,a)),I(h(f(a,b)))) \text{ with (2)}$$
$$\longrightarrow 0 \text{ with (1)}$$

In fact, rule (2) can be applied in arbitrary depth of a term which can be matched by the left hand side of rule (1):

$$f(h^n(f(a,b)),I(h^n(f(a,b)))) \longrightarrow f(h^n(f(b,a)),I(h^n(f(a,b)))) \text{ with (2)}$$
$$\longrightarrow 0 \text{ with (1)}$$

for $n = 1, 2, 3, \ldots$

There are several approaches to solve this problem:

- consider only term rewriting systems with linear rules [HUE 80]:
  If rule (2) were linear, then it can still be applied to $f(h(f(b,a)),I(h(f(a,b))))$:

  $$f(h(f(b,a)),I(h(f(a,b)))) \longrightarrow 0 \quad \text{with: } f(x,I(y)) \longrightarrow 0$$

  Problem:
  Many interesting problems require nonlinear rules

- Theory matching where the equational theory is generated by the cyclic rules [LB 77], [PS 81], [JK 84]

  Problem:
  For the computation of critical pairs a unification algorithm for the cyclic rules is necessary, but currently only few unfication algorithms are known, another problem is, that theory matching is NP-hard for nearly any interesting theory [KA 85]

- find a weaker property which implies confluence [GO 83]

  Problem:
  A term cannot be reduced only by the subrelation R(R), therfore we have to traverse the cycles. The length of the cycles depends on the length of the terms and the cyclic rules. For example: The number of elements in a cycle of a term t containing n commutative symbols, and where the commutativity is the only cyclic rule, is 2^n.

In this paper we present another approach, which can be applied for proofs in the initial algebra (inductive proofs). An equation $\alpha = \beta$ is inductive derivable from a set of equations E, if $\alpha = \beta$

holds in the initial algebra of E. This fact is provable by a consistency proof, the equation $\alpha = \beta$ holds in the initial algebra of E, if $E \cup \alpha = \beta$ does not identify more ground terms than E [MU 80], [HH 82]. For this method we will split the set of function symbols into defined functions and constructors. All constructor ground terms represent the elements from the domain and codomain of the defined functions. Every defined function should be completely defined in E, this means for every ground term t, there is a constructor ground term t', which is equal to t in the theory of E. We call a set of equations E consistent, iff all different constructor ground terms are not equal in the theory of E.
Here, we define a completeness and consistency property for term rewriting systems. A term rewriting system R is complete, iff every ground term is reducible to a constructor ground term by R. R is consistent, iff all constructor grond terms are in R-normal form. For a confluent term rewriting system R, the completeness (consistency) of R is equivalent to the completeness (consistency) of the equational theory induced by R.

(4.1) Definition
Let R be a term rewriting system.
- R is consistent, iff:
  $\forall\, t : t \in CGT \Longrightarrow t$ is in R-normal form
- R is complete, iff:
  $\forall\, t : t \in GT \Longrightarrow \exists\, t' : t \xrightarrow{*}_R t' \wedge t' \in CGT$

Note:
The completeness as defined here is different from the completeness of a term rewriting system R after applying the Knuth-Bendix Algorithm to R. In this paper we will use the meaning of Definition 4.1 for the completeness property.

The consistency test for a term rewriting systems is simple, it only has to be checked whether no left hand side of a rule is a constructor term. Then, no rule can be applied to a constructor ground term. The completeness of a terminating term rewriting system can be proven by the test of Kounalis [KO 85]. This test checks, whether every term of the form $f(t_1, \ldots, t_n)$, where f is a defined function and $t_1, \ldots, t_n$ are constructor ground terms, is reducible.
With these tests and the Knuth-Bendix Algorithm, we may try to prove, that a set of equations E' holds in the initial algebra of another set of equations E by the following method:

(1) Transform the equations from E into rewrite rules and complete them by the Knuth-Bendix Algorithm.
(2) Check the consistency and the completeness of the confluent term rewriting system, if it is inconsistent or incomplete stop with error.
(3) Add the equations from E' as rules to the term rewriting system and complete this extended set of rules by the Knuth-Bendix Algorithm
(4) Check the consistency of the extended term rewriting system, if R is consistent, then E' holds in the initial algebra of E, otherwise there are equations in E' which do not hold in the initial algebra of E.

The completeness test for this method works only for terminating term rewriting systems, but here we are interested in globally finite term rewriting systems. In the next lemma ,we will prove, that under certain conditions, it is sufficient to use the reducing subrelation for the completeness test. We get this reducing subrelation by splitting the set of rules into cyclic and reduction rules, as described in chapter 2.

(4.2) Lemma

Let R be a globally finite term rewriting system with:

- R is confluent and coherent
- R is consistent
- R = CR ∪ RR, CR are cyclic and RR are reduction rules

Then:

R is complete $\Longleftrightarrow$ RR is complete.

Proof

($\Longleftarrow$)

This direction is obvious, since RR is a subset of R

($\Longrightarrow$)

$t \in GT \Longrightarrow \exists\, t' : t \xrightarrow{*}_R t' \wedge t' \in CGT$ (R complete)

$\Longrightarrow$ t' is in R - normal form (R consistent)

$\Longrightarrow t{\downarrow}RR \xrightarrow{*}_{CR} t'$ (R coherent & confluent, Lemma 3.8)

$\Longrightarrow t' \xrightarrow{*}_{CR} t{\downarrow}RR$ (CR cyclic rules)

$\Longrightarrow t' = t{\downarrow}RR$ (t' is in R - normal form)

Now we will define a weaker relation $\longrightarrow\!\!\!\!\!\rightarrow$ on terms, where the variables from the left hand side of a rule can only be replaced by constructor terms (Definition 4.3). This restricted relation creates the same congruence on ground terms as the classical relation (Theorem 4.5). For consistent term rewriting systems, a rule cannot be applied to a subterm, which can be matched by a variable from another rule. Therefore, the coherence property may also hold for term rewriting systems with nonlinear rules. Theorem 4.7 gives a critcal pair test for the local confluence and the local coherence of the relation $\longrightarrow\!\!\!\!\!\rightarrow$.

(4.3) Definition

Let σ be a substitution, V a set of Variables and F = C ∪ D a set of functions, which can be splitted into defined functions D and constructors C. We call σ a constructor substitution, iff:

$$\forall\, x \in V : \sigma(x) \in CT$$

We denote the set of constructor substitution by Σ_C.

(4.4) Definition

Let R be a term rewriting system and F = C ∪ D the set of function symbols from R. We define the relation $\longrightarrow\!\!\!\!\!\rightarrow_R$ is as follows:

$t_1 \longrightarrow\!\!\!\!\!\rightarrow_R t_2 \Longleftrightarrow$:

$$\exists\, u : \exists\, \sigma \in \Sigma_C : \exists\, \alpha \longrightarrow \beta \in R : t_1 / u = \sigma(\alpha) \wedge t_2 = t_1\,[u \leftarrow \sigma(\beta)]$$

The $\longrightarrow\!\!\!\!\!\rightarrow_R$ relation is sometimes closer to the intended meaning of the rewrite rules, because we would like to define our functions on the elements of their domain, but not on arbitrary terms. The next lemma shows, that if a term rewriting system R is complete, then every ground term can be reduced to a constructor ground term by $\longrightarrow\!\!\!\!\!\rightarrow_R$.

(4.5) Lemma

Let R be a globally finite term rewriting system with:

- $R = CR \cup RR$, CR are cyclic rules and RR are reduction rules
- RR is complete

Then:

$$\forall\, t \in GT : \exists\, t' \in CGT : t \xrightarrow{*}_{RR} t'$$

Proof

We will proof, that for every ground term t, which is not a constructor ground term, a rule from RR can be applied. Since RR is terminating, RR will reduce t to a contructor ground term.

t is not a constructor ground term

$$\Longrightarrow \exists\, u : (t/u = f(t_1, \ldots, t_2) \wedge f \in D) \vee (t/u = c \wedge c \in D)$$

Let u be the deepest occurence of a defined function in t. Since RR is complete, there is a rule from RR, which reduces t / u:

$$\Longrightarrow \exists\, v : \exists\, \sigma : \exists\, \alpha \longrightarrow \beta \in RR : (t/u)/v = \sigma(\alpha) \quad \text{(RR complete)}$$

All occurences below u are constructor terms and if we assume, that no left hand side consists of a single variable we get:

$$\forall\, x \in V : \sigma(x) \in CGT$$

$$\Longrightarrow (t/u) \longrightarrow\!\!\!\!\!\rightarrow_{RR} (t/u)\,[v \leftarrow \sigma(\beta)]$$

$$\Longrightarrow t \text{ is reducible by } \longrightarrow\!\!\!\!\!\rightarrow_{RR}$$

The confluence of $\longrightarrow\!\!\!\!\!\rightarrow_R$ is sufficient for proving the equality of two ground terms by $\longrightarrow\!\!\!\!\!\rightarrow_R$:

(4.6) Theorem

Let R be a globally finite term rewriting system with:

- $R = CR \cup RR$, CR are cyclic rules and RR are reduction rules
- $\longrightarrow\!\!\!\!\!\rightarrow_R$ is confluent
- RR is complete

Then:

$$\forall\, t_1, t_2 \in GT : t_1 \overset{*}{\longleftrightarrow}_R t_2 \Longrightarrow \exists\, t : t_1 \xrightarrow{*}_R t \wedge t_2 \xrightarrow{*}_R t$$

Proof

We prove this theorem by induction on the length n of the minimal proof between t_1 and t_2 :

$$t_1 \overset{n}{\longleftrightarrow}_R t_2 \Longrightarrow \exists\, t : t_1 \xrightarrow{*}_R t \wedge t_2 \xrightarrow{*}_R t$$

The base case (n=0) is obvious, since t_1 is equal to t_2. For the proof of the induction step we assume:

$$t'_1 \overset{n-1}{\longleftrightarrow}_R t_2 \Longrightarrow \exists\, t : t'_1 \xrightarrow{*}_R t \wedge t_2 \xrightarrow{*}_R t \text{ (induction hypothesis)}$$

We have to prove:

$$t_1 \longleftrightarrow_R t'_1 \overset{n-1}{\longleftrightarrow}_R t_2 \Longrightarrow \exists\, t : t_1 \xrightarrow{*}_R t \wedge t_2 \xrightarrow{*}_R t$$

If $t_1 \vdash\!\!\!\rightarrow_R t'_1$, then there has to be a rule $\alpha \longrightarrow \beta$ or $\beta \longrightarrow \alpha$ from R and t'_1 is derivable from t_1 by $\alpha \longrightarrow \beta$, or t_1 is derivable by $\beta \longrightarrow \alpha$ from t'_1:

$$\exists u : \exists \sigma : t_1 / u = \sigma(\alpha) \wedge t'_1 = t_1 [u \leftarrow \sigma(\beta)]$$

All subterms in t_1 and t'_1, which are matched by variables from α and β can be reduced by $\overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R$ to constructor ground terms (R complete), then the rule $\alpha \longrightarrow \beta$ or $\beta \longrightarrow \alpha$ can be applied to the reduced term of t_1 or t'_1:

We define σ':

- $x \in D(\sigma)$: $\sigma'(x) = t$ and t is the normal form of $\sigma(x)$ in $\overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R$
- $x \notin D(\sigma)$: $\sigma'(x) = x$

R is complete:

$\Longrightarrow \forall x \in D(\sigma') : \sigma'(x) \in CGT$ with : Lemma 4.5

$\Longrightarrow \sigma' \in \Sigma_C$

Then we get:

$$\sigma(\alpha) \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R \sigma'(\alpha) \wedge \sigma(\beta) \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R \sigma'(\beta)$$

$$\Longrightarrow t_1 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t_1 [u \leftarrow \sigma'(\alpha)] \wedge t_1 [u \leftarrow \sigma(\beta)] \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t_1 [u \leftarrow \sigma'(\beta)]$$

There are two cases:

(i) $\alpha \longrightarrow \beta \in R$: $t_1 [u \leftarrow \sigma'(\alpha)] \longrightarrow\!\!\!\!\rightarrow_R t_1 [u \leftarrow \sigma'(\beta)]$

(ii) $\beta \longrightarrow \alpha \in R$: $t_1 [u \leftarrow \sigma'(\beta)] \longrightarrow\!\!\!\!\rightarrow_R t_1 [u \leftarrow \sigma'(\alpha)]$

For case (i) we let $t_0 = t_1 [u \leftarrow \sigma'(\beta)]$ and for case (ii) we let $t_0 = t_1 [u \leftarrow \sigma'(\alpha)]$. Then we get for both cases:

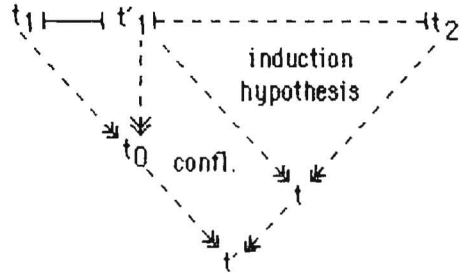
$$t_1 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t_0 \wedge t'_1 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t_0$$

with the induction hypothesis: $t'_1 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t \wedge t_2 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t$ and the confluence of $\overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R$ we get:

$\exists t' : t \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t' \wedge t_0 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t'$

$\Longrightarrow t_1 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t' \wedge t_2 \overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R t'$

The following diagram summmarizes the proof of the induction step:



In the next theorem, we show how the local confluence and the local coherence of $\overset{*}{\longrightarrow\!\!\!\!\rightarrow}_R$ can be checked by critical pair tests. In this theorem, we denote the normal form of a term t with respect to $\longrightarrow\!\!\!\!\rightarrow_R$ also by $t{\downarrow}R$.

(4.7) Theorem

Let R = CR ∪ RR be a globally finite and consistent term rewriting system, which can be splitted into cyclic rules CR and reduction rules RR.

R is locally confluent if condition (1) is satisfied:

(1) $\forall\, \alpha \longrightarrow \beta, \gamma \longrightarrow \delta \in RR\ \forall\, u \in O(\alpha)\ \forall\, \sigma \in \Sigma_C : \sigma(\alpha) / u = \sigma(\gamma)$

$\Longrightarrow \sigma(\alpha)[u \leftarrow \sigma(\delta)]\downarrow RR \xrightarrow{*}_{CR} \sigma(\beta)\downarrow RR$

R is locally coherent if condition (2) and (3) are satisfied:

(2) $\forall\, \alpha \longrightarrow \beta \in RR, \gamma \longrightarrow \delta \in CR\ \forall\, u \in O(\alpha)\ \forall\, \sigma \in \Sigma_C : \sigma(\alpha) / u = \sigma(\gamma)$

$\Longrightarrow \exists\, t' : \sigma(\alpha)[u \leftarrow \sigma(\delta)] \longrightarrow\!\!\!\!\!\rightarrow_{RR} t' \wedge t'\downarrow RR \xrightarrow{*}_{CR} \sigma(\beta)\downarrow RR$

(3) $\forall\, \alpha \longrightarrow \beta \in RR, \gamma \longrightarrow \delta \in CR\ \forall\, u \in O(\gamma)\ \forall\, \sigma \in \Sigma_C : \sigma(\alpha) = \sigma(\gamma) / u$

$\Longrightarrow \exists\, t' : \sigma(\delta) \longrightarrow\!\!\!\!\!\rightarrow_{RR} t' \wedge t'\downarrow RR \xrightarrow{*}_{CR} \sigma(\gamma)[u \leftarrow \sigma(\beta)]\downarrow RR$

If R is confluent and coherent, then the conditions (1), (2) and (3) are satisfied.

Proof

The details of this proof are very similar to the proof for the classical critical pair test, therefore we will sketch only the main ideas.

R is locally confluent

Let t_1 be derivable from t by $\alpha \longrightarrow \beta \in R$ and t_2 be derivable from t by $\gamma \longrightarrow \delta \in R$. If $\alpha \longrightarrow \beta \in CR$, then t is derivable from t_1 and t_2 is in the reduction set of t_1. The same argument holds, if $\gamma \longrightarrow \delta \in CR$. Otherwise we get two cases:

(i) $\alpha \longrightarrow \beta$ and $\gamma \longrightarrow \delta$ are applied at disjoint occurences.
Both rules can be applied simultaneously to t and we get a term t', which is in the reduction set of t_1 and t_2.

(ii) $\alpha \longrightarrow \beta$ and $\gamma \longrightarrow \delta$ are applied at overlapping occurences.
No rule can be applied to a subterm, which can be matched by a variable of another rule (R consistent), therefore we get a critical overlapping between these rules. For this critical overlapping we have to consider all critical pairs between $\alpha \longrightarrow \beta$ and $\gamma \longrightarrow \delta$.

R is locally coherent

Let t_1 be derivable from t by $\alpha \longrightarrow \beta \in RR$ and t_2 be derivable from t by $\gamma \longrightarrow \delta \in CR$.

If $\alpha \longrightarrow \beta$ and $\gamma \longrightarrow \delta$ are applied at disjoint occurences, then t_2 can be reduced by $\alpha \longrightarrow \beta$ to t' and t' is derivable from t_1. Otherwise $\alpha \longrightarrow \beta$ is applied at an occurence below $\gamma \longrightarrow \delta$ (i) or $\gamma \longrightarrow \delta$ is applied below $\alpha \longrightarrow \beta$ (ii). For case (i) we have to consider critical pairs as in condition (3) and for case (ii) we consider critical pairs as in condition (2).

(1),(2) and (3) are satisfied if R is confluent and coherent

Every condition is obviously satisfied in a confluent and coherent term rewriting system, therefore we skip this proof.

(5) Extended Knuth-Bendix Procedure

A simple completion procedure, which bases on the results of this paper may look as follows:

Completion Procedure

Input: - A set of equations E
- A weak term ordering $\succeq$

(1) Create rules from E:

$CR := \{ \alpha \longrightarrow \beta \mid (\alpha = \beta \in E \vee \beta = \alpha \in E) \wedge \alpha \succeq \beta \wedge \beta \succeq \alpha \}$
$RR := \{ \alpha \longrightarrow \beta \mid (\alpha = \beta \in E \vee \beta = \alpha \in E) \wedge \alpha \succeq \beta \wedge \neg \beta \succeq \alpha \}$

$\exists \alpha = \beta \in E : \alpha \longrightarrow \beta \notin R \wedge \beta \longrightarrow \alpha \notin R$?
true: stop with failure
false: continue with (2)

(2) Check local coherence and local confluence:

(i) $\forall \alpha \longrightarrow \beta, \gamma \longrightarrow \delta \in RR \, \forall u \in O(\alpha) \, \forall \sigma \in \Sigma_C : \sigma(\alpha) / u = \sigma(\gamma)$
$\Longrightarrow \sigma(\alpha)[u \leftarrow \sigma(\delta)]\downarrow RR \xrightarrow{*}_{CR} \sigma(\beta)\downarrow RR$

(ii) $\forall \alpha \longrightarrow \beta \in RR, \gamma \longrightarrow \delta \in CR \, \forall u \in O(\alpha) \, \forall \sigma \in \Sigma_C : \sigma(\alpha) / u = \sigma(\gamma)$
$\Longrightarrow \exists t' : \sigma(\alpha)[u \leftarrow \sigma(\delta)] \longrightarrow_{RR} t' \wedge t'\downarrow RR \xrightarrow{*}_{CR} \sigma(\beta)\downarrow RR$

(iii) $\forall \alpha \longrightarrow \beta \in RR, \gamma \longrightarrow \delta \in CR \, \forall u \in O(\gamma) \forall \sigma \in \Sigma_C : \sigma(\alpha) = \sigma(\gamma) / u$
$\Longrightarrow \exists t' : \sigma(\delta) \longrightarrow_{RR} t' \wedge t'\downarrow RR \xrightarrow{*}_{CR} \sigma(\gamma)[u \leftarrow \sigma(\beta)]\downarrow RR$

If (i), (ii) or (iii) do not hold then continue with step (3), otherwise stop with success.

(3) There are three cases:

- $\exists \alpha \longrightarrow \beta, \gamma \longrightarrow \delta \in RR \, \exists u \in O(\alpha) \, \exists \sigma \in \Sigma_C : \sigma(\alpha) / u = \sigma(\gamma)$
$\Longrightarrow \sigma(\alpha)[u \leftarrow \sigma(\delta)]\downarrow RR \not\xrightarrow{*}_{CR} \sigma(\beta)\downarrow RR$
$\longrightarrow$ continue with step (3)

- $\exists \alpha \longrightarrow \beta \in RR, \gamma \longrightarrow \delta \in CR \, \exists u \in O(\alpha) \, \exists \sigma \in \Sigma_C : \sigma(\alpha) / u = \sigma(\gamma)$
$\Longrightarrow \forall t' : \sigma(\alpha)[u \leftarrow \sigma(\delta)] \not\longrightarrow_{RR} t' \vee t'\downarrow RR \not\xrightarrow{*}_{CR} \sigma(\beta)\downarrow RR$
$\longrightarrow$ add $\sigma(\alpha)[u \leftarrow \sigma(\delta)] \longrightarrow \sigma(\beta)\downarrow RR$ to RR and continue with (2)

- $\exists \alpha \longrightarrow \beta \in RR, \gamma \longrightarrow \delta \in CR \, \exists u \in O(\gamma) \exists \sigma \in \Sigma_C : \sigma(\alpha) = \sigma(\gamma) / u$
$\Longrightarrow \forall t' : \sigma(\delta) \not\longrightarrow_{RR} t' \vee t'\downarrow RR \not\xrightarrow{*}_{CR} \sigma(\gamma)[u \leftarrow \sigma(\beta)]\downarrow RR$
$\longrightarrow$ add $\sigma(\delta) \longrightarrow \sigma(\gamma)[u \leftarrow \sigma(\beta)]\downarrow RR$ to RR and continue with (2)

(3) There are four cases:

- $\sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR} \geq \sigma(\beta)\downarrow_{RR} \wedge\ \sigma(\beta)\downarrow_{RR} \geq \sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR}$
  add $\sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR} \longrightarrow \sigma(\beta)\downarrow_{RR}$ to CR
  add $\sigma(\beta)\downarrow_{RR} \longrightarrow \sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR}$ to CR
  continue with (2)

- $\sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR} \geq \sigma(\beta)\downarrow_{RR} \wedge\ \neg\, \sigma(\beta)\downarrow_{RR} \geq \sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR}$
  add $\sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR} \longrightarrow \sigma(\beta)\downarrow_{RR}$ to RR
  continue with (2)

- $\sigma(\beta)\downarrow_{RR} \geq \sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR} \wedge\ \neg\, \sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR} \geq \sigma(\beta)\downarrow_{RR}$
  add $\sigma(\beta)\downarrow_{RR} \longrightarrow \sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR}$ to RR
  continue with (2)

- $\sigma(\alpha)\,[u \leftarrow \sigma(\delta)]\downarrow_{RR}$ and $\sigma(\beta)\downarrow_{RR}$ are not comperable
  stop with failure

For reasons of efficiency we had to implement an extended version of this completion procedure:

- The completion procedure should be able to remove rules during the completion, if they can be reduced by new rules. After removing a rule from a set R, some critical pairs which were reducible by R might no longer be reducible by the new system. Then the local coherence or local confluence for these pairs may not hold. For this purpose, we have proven that weak versions of the local confluence and local coherence property still imply the confluence. These weaker properties are guaranteed for all critical pairs, which have been tested during the completion, even if some rules have been removed from R.

- The relation $\longrightarrow\!\!\!\!\!\!\rightarrow_R$ is weaker than the relation $\longrightarrow_R$, therefore some critical pairs which are reducible by $\longrightarrow_R$ might not be reducible by $\longrightarrow\!\!\!\!\!\!\rightarrow_R$. This will cause the completion procedure to diverge sometimes, when it would converge with the relation $\longrightarrow_R$. In our implementation we distinguish between declared variables and other variables. A declared variable can only be replaced by a constructor term and the other variables may be replaced by arbitrary terms. This extension works correctly, if all nonlinear variables in left hand sides of rules are declared.

For a confluent and globally finite term rewriting system, the completeness (Definition 4.1) of the reduction rules can be checked by the test of Kounalis.

In the appendix we give an example, which was completed by our implementation. This example contains a nonlinear rule and some cyclic rules, for example:

$$if(p,if(q,x,y),if(q,u,v)) = if(q,if(p,x,u),if(p,y,v))$$

This kind of equations are often created during the completion of term rewriting systems with axioms for conditions. Currently, no unification algorithm is known for this axiom, therefore an approach which bases on theory unification, cannot be applied to this theory.

(6) Conclusion

There are two aspects for the results of this paper:
Firstly, it desrcibes a extended completion procedure for inductive proofs. This procedure is able to complete globally finite term rewriting systems with arbitrary cyclic rules, if these rules cause finite cycles. No theory unification is required, nor does it restrict the kind of rules (linear rules, certain cyclic rules, . . .). It works also efficient, since no theory matching is required for reducing terms and we have to traverse the cycles only for comparing normal forms, which are usually small terms.
Secondly, it describes a theoretical framework for extending the Knuth-Bendix Algorithm to globally finite term rewriting systems, which differs from former approaches. These former approaches allowed reductions modulo equivalence classes (e.g. [JK 84]). This keeps the finite termination property, but requires a theoretical background for this extension, where we have to consider two different relations. This complicates sometimes the theory, because we have to consider both relations and their combinations. Another problem is, that all cyclic rules have to be incorporated into the matcher and unfier of the completion algorithm. In our theoretical frame, the confluence results are completely independent from the kind of rules, we may use a theory matcher, where the theory is generated by an arbitrary subset of the cyclic rules.

References

[DE 82] Dershowitz N.:
Orderings for Term-Rewriting Systems
Theoretical Computer Science 17 p. 279-301
North-Holland Publishing Company (1982)

[GO 83] Göbel, R.:
A Completion Procedure for Globally Finite Term Rewriting Systems
Proceedings of an NSF Workshop on the Rewrite Rule Laboratory
General Electric
Schenectady (1983)

[HH 82] Huet G., Hullot J.:
Proofs by Induction in Equational Theories with Constructors
Journal of the Association for Computing Machinery 25(2), p. 239-266 (1982)

[HO 80] Huet, G., Oppen D.:
Equations and Rewrite Rules: A Survey
Technical Report CSL - 111
SRI International (1980)

[HU 77] Huet, G.:
Confluent Reductions: Abstract Properties and Applications to Term Rewriting Systems
18th IEEE Symposium on foundations of Computer Science, p. 30 - 45 (1977)

[JK 84] Jouannaud J. P., Kirchner H.:
Completion of a Set of Rules Modulo a Set of Equations
11th Symposium on Principles of Programming Languages
Salt Lake City, Utah (1984)

[KA 85] Kapur D., Narendran P., Benanav D.:
Complexity of Matching Problems
First International Conference on Rewriting Techniques and Applications
Dijon (1985)

[KB 70] Knuth D., Bendix P.:
Simple Word Problems in Universal Algebras
Computational Problems in Abstract Algebra
Ed. Leech I., Pergamon Press, p. 263 - 297 (1970)

[KO 85] Kounalis E.:
Completeness in Data Type Specifications
Proc. 3rd EUROCAL
Linz (1985)

[LB 77] Lankford D.S., Ballantyne A.M.:
Decision Procedure for Simple Equational Theories with Commutative-Associative Axioms : Complete Sets of Permutative Reductions
Report ATP - 39
Departments of Mathematics and Computer Science
University of Texas at Austin (1977)

[MU 80] Musser D.:
On Proving Inductive Properties of Abstract Data Types
7th Annual ACM Symposium on Principles of Programming Languages
Las Vegas (1980)

[NM 84] Narendran P., McNaughton R.:
The Undecidability of the Preperfectness of Thue Systems
Theoretical Computer Science 31, p. 164-174
North Holland 1984

[PS 81] Stickel M., Peterson G.:
Complete Sets of Reductions for Some Equational Theories
Journal of the ACM, Vol 28, No. 2, p. 233-264 (1981)

Appendix

These equations define elementary functions on natural numbers. The natural numbers are represented by ground terms containg the constructors S (successor) and 0 (zero). The functions ADD (addition), SUB (subtraction) and MUL (multiplication) are directly defined on these constructors. The definition of the function DIV (divide) requires axioms for the predicate LESS, an IF-THEN-ELSE construct and two new constructors TRUE and FALSE:

```
ADD(0,x) = x
ADD(S(x),y) = S(ADD(x,y))

SUB(0,x) = 0
SUB(x,0) = x
SUB(S(x),S(y)) = SUB(x,y)

MUL(0,x) = 0
MUL(S(x),y) = ADD(MUL(x,y),y)

LESS(x,0) = FALSE
LESS(0,S(x)) = TRUE
LESS(S(x),S(y)) = LESS(x,y)

IF(TRUE,x,y) = x
IF(FALSE,x,y) = y

DIV(x,0) = 0
DIV(0,x) = 0
DIV(S(x),S(y)) = IF(LESS(x,y),0,S(DIV(SUB(x,y),S(y))))
```

We want to prove the following equations:

```
ADD(x,y) = ADD(y,x)
ADD(ADD(x,y),z) = ADD(x,ADD(y,z))
SUB(x,x) = 0
MUL(ADD(x,y),z) = ADD(MUL(x,z),MUL(y,z))
IF(p,IF(q,x,y),IF(q,u,v)) = IF(q,IF(p,x,u),IF(p,y,v))
```

The set of axioms is complete and consistent and we start the extended completion procedure with the union of both sets. The variable x in SUB(x,x) is not linear, therefore we have to declare this variable. In the protocol on the next page, declared variables are prefixed by a c_.
The completion procedure creates two new rules and then stops with success.

PROTOCOL OF COMPLETION

(1): ADD(0,X) --> X	(reduction rule)
(2): SUB(X,0) --> X	(reduction rule)
(3): SUB(0,X) --> 0	(reduction rule)
(4): LESS(X,0) --> FALSE	(reduction rule)
(5): MUL(0,X) --> 0	(reduction rule)
(6): DIV(X,0) --> 0	(reduction rule)
(7): DIV(0,X) --> 0	(reduction rule)
(8): SUB(c_x,c_x) --> 0	(reduction rule)
(9): ADD(Y,X) --> ADD(X,Y)	(cyclic rule)
9,1 ==> (10): ADD(X,0) --> X	(reduction rule)
(11): ADD(S(X),Y) --> S(ADD(X,Y))	(reduction rule)
(12): LESS(0,S(x)) --> TRUE	(reduction rule)
(13): IF(TRUE,X,Y) --> X	(reduction rule)
(14): IF(FALSE,X,Y) --> Y	(reduction rule)
9,11 ==> (15): ADD(X,S(Y)) --> S(ADD(Y,X))	(reduction rule)
(16): SUB(S(X),S(Y)) --> SUB(X,Y)	(reduction rule)
(17): LESS(S(X),S(Y)) --> LESS(X,Y)	(reduction rule)
(18): MUL(S(X),Y) --> ADD(MUL(X,Y),Y)	(reduction rule)
(19): ADD(X,ADD(Y,Z)) --> ADD(ADD(X,Y),Z)	(cyclic rule)
(20): MUL(ADD(X,Y),Z) --> ADD(MUL(X,Z),MUL(Y,Z))	(reduction rule)
(21): IF(Y,IF(X,Z,V),IF(X,U,W)) --> IF(X,IF(Y,Z,U),IF(Y,V,W))	(cyclic rule)
(22): DIV(S(X),S(Y)) --> IF(LESS(X,Y),0,S(DIV(SUB(X,Y),S(Y))))	(reduction rule)

The system is complete !

FINAL RULES

cyclic rule:

(9): ADD(Y,X) --> ADD(Y,X)
(19): AD(X,ADD(Y,Z)) --> ADD(ADD(X,Y),Z)
(21): IF(Y,IF(X,Z,V),IF(X,U,W)) --> IF(X,IF(Y,Z,U),IF(Y,V,W))

reduction rules:

(1): ADD(0,X) --> X
(2): SUB(X,0) --> X
(3): SUB(0,X) --> 0
(4): LESS(X,0) --> FALSE
(5): MUL(0,X) --> 0
(6): DIV(X,0) --> 0
(7): DIV(0,X) --> 0
(8): SUB(c_x,c_x) --> 0
9,1 ==> (10): ADD(X,0) --> X
(11): ADD(S(X),Y) --> S(ADD(X,Y))
(12): LESS(0,S(x)) --> TRUE
(13): IF(TRUE,X,Y) --> X
(14): IF(FALSE,X,Y) --> Y
9,11 ==> (15): ADD(X,S(Y)) --> S(ADD(X,Y))
(16): SUB(S(X),S(Y)) --> SUB(X,Y)
(17): LESS(S(X),S(Y)) --> LESS(X,Y)
(18): MUL(S(X),Y) --> ADD(MUL(X,Y),Y)
(20): MUL(ADD(X,Y),Z) --> ADD(MUL(X,Z),MUL(Y,Z))
(22): DIV(S(X),S(Y)) --> IF(LESS(X,Y),0,S(DIV(SUB(X,Y),S(Y))))