A Constructor-Based Approach for Positive/Negative-Conditional Equational Specifications

Claus-Peter Wirth, Bernhard Gramlich

Fachbereich Informatik, Universität Kaiserslautern,
Postfach 3049, W-6750 Kaiserslautern, Germany
{wirth, gramlich}@informatik.uni-kl.de

Abstract: We present a constructor-based approach for assigning appropriate semantics to algebraic specifications given by finite sets of positive/negative-conditional equations. Under the assumption of confluence of the reduction relation we define, the factor algebra of the ground term algebra modulo the congruence of this reduction relation is a minimal model which is (beyond that) the minimum of all models that do not identify more constructor ground terms than necessary.

1 Introduction and Overview

We present a new uniform (i. e. not using "built-in"s or specification hierarchies) constructor-based approach for assigning appropriate semantics to algebraic specifications with finite sets of positive/negative-conditional ("p/n-conditional" for short) equations. A reduction relation is defined which allows us to generalize the fundamental results for positive-conditional reduction systems (cf. [5, 3]). Under the assumption of confluence, the factor algebra of the ground term algebra modulo the congruence of our reduction relation is an (up to isomorphism) uniquely determined minimal model, which (as developed in Wirth[8]) can be used to establish inductive validity and allows a straightforward abstraction for the treatment of non-ground equational clauses by contextual rewriting.

The class of models of a finite set R of p/n-conditional equations does not contain in general a minimum model (w. r. t. homomorphic mapping). The most promising attempt in literature to overcome this problem has been that in Kaplan[6]. There, one of the minimal ("quasi-initial") models is distinguished from the other minimal models by the use of control information extracted from the rules, which must be compatible with a noetherian ordering. In addition, Kaplan gives a straightforward ground term reduction relation. However, the approach of Kaplan[6] violates the paradigm of separation of logic and control and does not allow to express the distinction of the minimal model without the control part of the specification. Therefore, one loses the nice aspects of the pure logic view on the specification which becomes a mere program instead (cf. sect. 3). For that reason, we choose a new different approach. Instead of using control information we introduce two syntactic restrictions: Firstly, for

This research was supported by the Deutsche Forschungsgemeinschaft, SFB 314 (D4-Projekt)

a condition to be fulfilled we require the evaluation values of the terms of the negative equations to be in the set of evaluation values of constructor ground terms by adding condition literals expressing this property. Secondly, we restrict the constructor rules (which express equalities among the constructor terms) to have "Horn"-form and to be "constructor-preserving". We can then carry over the results of Kaplan[6] without using control information or noetherian orderings anymore (cf. sect. 5, 6). As a consequence, our reduction relation does not need to be noetherian or to be restricted to ground terms. Contrary to [6], we can show the monotony of this reduction relation w. r. t. the extension of the specification. As in [6], under the assumption of confluence of our reduction relation, the factor algebra of the ground term algebra modulo the congruence of our reduction relation is a minimal model for our specification. Unlike [6], however, it is also the minimum of all models that do not identify more constructor ground terms than necessary. To achieve decidability of reducibility we define several kinds of compatibility of R with a reduction ordering and present a complete critical-pair test à la Knuth-Bendix for the confluence of our reduction relation. The proofs, further concepts, and a deeper discussion can be found in Wirth&Gramlich[9].

2 Basic Notions and Notations

We will consider terms of fixed arity over a *many-sorted signature* $\mathrm{sig} = (\mathrm{F}, \mathrm{S}, \alpha)$ consisting of an enumerable set of function symbols F, a finite set of sorts S (disjoint from F), and a computable arity-function $\alpha : \mathrm{F} \rightarrow \mathrm{S}^{+}$. F has a decidable subset C of *constructor symbols*, which constitute their own signature $\mathrm{cons} = (\mathrm{C}, \mathrm{S}, \alpha|_{\mathrm{C}})$. We assume a fixed S-sorted variable-system V with disjoint infinite decidable ranges only. For the family of variables occurring in A we will use $\mathcal{V}(A)$. $\mathcal{T}(\mathrm{sig})$ (resp. $\mathcal{T}(\mathrm{cons})$) denotes the S-sorted family of all (well-sorted) *terms* over sig and V (resp. *constructor terms*), while $\mathcal{GT}(\mathrm{sig})$ (resp. $\mathcal{GT}(\mathrm{cons})$) denotes the S-sorted family of all *ground terms* over sig (resp. *constructor ground terms*). We assume $\forall s \in \mathrm{S} : \mathcal{GT}(\mathrm{cons})_s \neq \emptyset$. For a term t we denote by $\mathcal{O}(t)$ the set of its *occurrences*, by t/p the subterm of t at occurrence p, and by $t[\,p \leftarrow t'\,]$ the result of replacing t/p with t' in t. The set of *substitutions* from a variable-system X to an S-sorted family of sets T is denoted by $\mathcal{SUB}(X,T)$. A relation R on $\mathcal{T}(\mathrm{sig})$ is called: *stable (w. r. t. substitution)* $\underline{\text{:iff}}$ $\forall (t,t') \in R : \forall \sigma \in \mathcal{SUB}(\mathrm{V}, \mathcal{T}(\mathrm{sig})) : (t\sigma, t'\sigma) \in R$; *monotonic (w. r. t. replacement)* $\underline{\text{:iff}}$ $\forall (t',t'') \in R : \forall t \in \mathcal{T}(\mathrm{sig}) : \forall p \in \mathcal{O}(t) : \forall s \in \mathrm{S} : (t/p, t', t'' \in \mathcal{T}(\mathrm{sig})_s \;\Rightarrow\; (t[\,p \leftarrow t'\,], t[\,p \leftarrow t''\,]) \in R)$. A relation R is called: *noetherian* $\underline{\text{:iff}}$ there is no $a : \mathbb{N} \rightarrow \mathrm{dom}(R)$ with $\forall i \in \mathbb{N} : (a_i, a_{i+1}) \in R$. An ordering $<$ or $>$ is called *well-founded* $\underline{\text{:iff}}$ $>$ is noetherian. A *congruence* is a monotonic equivalence. The symmetric closure of a relation $\Longrightarrow$ will be denoted by $\Longleftrightarrow$; its transitive closure by $\stackrel{\oplus}{\Longrightarrow}$; its reflexive transitive closure by $\stackrel{\circledast}{\Longrightarrow}$. Two terms v, w are called *joinable w. r. t.* $\Longrightarrow$ $\underline{\text{:iff}}$ $v \downarrow w$ $\underline{\text{:iff}}$ $v \stackrel{\circledast}{\Longrightarrow} \circ \stackrel{\circledast}{\Longleftarrow} w$. $\Longrightarrow$ is called *confluent below* u $\underline{\text{:iff}}$ $\forall v, w : ((v \stackrel{\oplus}{\Longleftarrow} u \stackrel{\oplus}{\Longrightarrow} w) \;\Rightarrow\; (v \downarrow w))$; it is called *confluent* $\underline{\text{:iff}}$ it is confluent below all u. When $\Longrightarrow$ is a relation on $\mathcal{T}(\mathrm{sig})$, we say $\Longrightarrow$ to be *ground* ... $\underline{\text{:iff}}$ $\Longrightarrow \cap\, (\mathcal{GT}(\mathrm{sig}) \times \mathcal{GT}(\mathrm{sig}))$ is ....

3 Motivation

Kaplan[6] defines a [negative] equation in the condition of a p/n-conditional equation to hold <u>:iff</u> its terms [do not] have a common reduct. If the resulting reduction relation is confluent and the rules are *decreasing* (cf. [3]), then its congruence closure is minimal (w. r. t. '$\subseteq$') among the congruence relations whose factor algebra (w. r. t. $\mathcal{GT}(\mathrm{sig})$) is a model of R. Confluence is essential here: Firstly, for the correctness of testing semantic equality of two condition terms by looking for a common reduct. Secondly, for the congruence to be minimal (cf. [9]). While confluence can be dropped for merely positive conditional equations by testing for congruence instead for a common reduct, it is worse for p/n-conditional equations: As any congruence yielding a model of the rules $c = d \longleftarrow d \neq e$ and $c = e$ must contain (d, e), the test of $d \neq e$ by such a congruence will always fail, so that we cannot establish $c = d$ by testing the condition of the first rule. But the rules have the minimum model '$c = d = e$', which cannot be obtained by condition-testing anymore, but only by paramodulation and factoring instead.

The major shortcoming of the reduction relation of [6], however, is that its congruence closure is not a minimum (smaller than anything else) but only minimal (there is nothing smaller) among the congruences yielding a model of R: There might be reductions $s \Longrightarrow t$ with $s = t$ not holding in all models logically specified by R. Kaplan[6] argues as follows: By writing $c = d \longleftarrow d \neq e$ instead of the logically equivalent $c = d \vee d = e$ the specifier adds some "operational" information to control the choice of the intended minimal congruence '$c = d$' of the congruences yielding a model of R ('$c = d$', '$d = e$', and '$c = d = e$'). This means that semantics is given by control and not expressible without; thereby violating the paradigm of separation of logic and control: It is not a *logical* specification (suitable for computation) anymore (as it was the case with positive conditional equations), but a program with none but operational semantics. Therefore, we lose the nice aspects of the pure logic view, e. g. the monotony of logic is lost:

Example 3.1 R: $x - 0 = x$; $\mathsf{s}(x) - \mathsf{s}(y) = x - y$;
$\mathsf{memberp}(x, \mathsf{nil}) = \mathsf{false}$; $\mathsf{memberp}(x, \mathsf{cons}(y, l)) = \mathsf{true} \longleftarrow x = y$;
$\mathsf{memberp}(x, \mathsf{cons}(y, l)) = \mathsf{memberp}(x, l) \longleftarrow x \neq y$.
We have $\mathsf{memberp}(0, \mathsf{cons}(0 - \mathsf{s}(0), \mathsf{nil})) \overset{\oplus}{\Longrightarrow} \mathsf{false}$ which no longer holds after adding the rule $0 - \mathsf{s}(x) = 0$.

Example 3.2 (Minimal Non-Minimum Models in Practice of Specification) (continuing Example 3.1)
We exclude the function symbols $\mathsf{0}$, s, and $-$, and enrich the signature with two constants a, b. Consider the following two congruence relations on ground terms, given by their congruence classes:
$\underline{\overset{\oplus}{\Longleftrightarrow}}$: $\{\mathsf{a}\}$; $\{\mathsf{b}\}$; $\{\mathsf{false}\} \cup \{\mathsf{memberp}(x, l) | (x \in \{\mathsf{a}, \mathsf{b}\} \wedge (x \text{ doesn't occur in } l))\}$;
$\{\mathsf{true}\} \cup \{\mathsf{memberp}(x, l) | (x \in \{\mathsf{a}, \mathsf{b}\} \wedge (x \text{ does occur in } l))\}$;
$\{\mathsf{nil}\}$; $\{\mathsf{cons}(\mathsf{a}, \mathsf{nil})\}$; $\{\mathsf{cons}(\mathsf{b}, \mathsf{nil})\}$; $\{\mathsf{cons}(\mathsf{a}, \mathsf{cons}(\mathsf{a}, \mathsf{nil}))\}$; $\{\mathsf{cons}(\mathsf{a}, \mathsf{cons}(\mathsf{b}, \mathsf{nil}))\}$; ...
$\underline{\sim}$: $\{\mathsf{a}, \mathsf{b}\}$; $\{\mathsf{false}, \mathsf{memberp}(\mathsf{a}, \mathsf{nil}), \mathsf{memberp}(\mathsf{b}, \mathsf{nil})\}$;
$\{\mathsf{true}\} \cup \{\mathsf{memberp}(x, l) | (x \in \{\mathsf{a}, \mathsf{b}\} \wedge l \neq \mathsf{nil})\}$;
$\{\mathsf{nil}\}$; $\{\mathsf{cons}(\mathsf{a}, \mathsf{nil}), \mathsf{cons}(\mathsf{b}, \mathsf{nil})\}$; $\{\mathsf{cons}(x, \mathsf{cons}(y, \mathsf{nil})) \mid x, y \in \{\mathsf{a}, \mathsf{b}\}\}$; ...

Now, both $\overset{\oplus}{\Longleftrightarrow}$ and $\sim$ yield a model of R. By $\mathsf{a}\sim\mathsf{b}$; $\mathsf{a}\overset{\oplus}{\not\Longleftrightarrow}\mathsf{b}$; we know that $\sim$ is not a minimum. Considering $\mathsf{memberp}(\mathsf{a}, \mathsf{cons}(\mathsf{b}, \mathsf{nil}))$ and false, we know that $\overset{\oplus}{\Longleftrightarrow}$ isn't a minimum either. But both are minimal among the congruences that yield a model of R. Hence their intersection does not yield a model of R. Thus, we have to choose between $\mathsf{a} \neq \mathsf{b}$ and $\mathsf{memberp}(\mathsf{a}, \mathsf{cons}(\mathsf{b}, \mathsf{nil})) \neq \mathsf{false}$. As $\overset{\oplus}{\Longleftrightarrow}$ is somehow more appealing than $\sim$, one may argue that $\mathsf{a} \neq \mathsf{b}$ is somewhat more important than $\mathsf{memberp\ a\ cons\ b\ nil} \neq \mathsf{false}$ by stating a, b to be constructors and thinking freeness of constructors to be more important than that of non-constructors. But this treatment does not solve the problem in general: If

1. a or b is changed into a (composite) non-constructor term,

or

2. $\mathsf{memberp}$ is stated to be a constructor symbol too,

then the very same problem arises again.

Now, while the simple attempt above fails, the intended bias towards freeness of constructor terms can be achieved the following way:

A. Adding condition literals (cf. sect. 4) expressing *definedness* for all terms of negative equations in the condition. A term t is *defined* :iff 'Def t' holds :iff t has a congruent constructor ground term. For our example above this means that the last $\mathsf{memberp}$-rule is not applicable if a or b is undefined, thereby avoiding the problem of (1) above.

B. Forcing rules whose left-hand sides are constructor terms to have no negative equations in their conditions and to be constructor-preserving[1]. For our example above this means that $\mathsf{memberp}$ cannot be a constructor symbol, thereby avoiding the problem of (2) above.

If a, b are defined terms, then $\overset{\oplus}{\Longleftrightarrow}$ becomes the *minimum* congruence of those congruences that do not identify more defined terms than necessary. Contrariwise, if a or b is undefined, then the intersection of $\overset{\oplus}{\Longleftrightarrow}$ and $\sim$ becomes a model because the last $\mathsf{memberp}$-rule now reads

$$\text{`}\mathsf{memberp}(x, \mathsf{cons}(y, l)) = \mathsf{memberp}(x, l) \longleftarrow x \neq y,\ \mathrm{Def}\, x, \mathrm{Def}\, y\text{'};$$

thus, $\mathsf{memberp}(\mathsf{a}, \mathsf{cons}(\mathsf{b}, \mathsf{nil}))$ is undefined now. (B) is purely syntactic and not very restrictive in practice as it only limits congruences between constructor terms (and this even less restrictively than usual). (A) is not a usage of control information as before. It just means that '$\neq$' is syntactically restricted to defined terms. Undefined terms are due to partially specified functions or incomplete knowledge about the model world. They are often thought to be equal to some unknown defined terms (cf. Kapur&Musser[7]). Based on this tradition, our approach can be justified the following way: If two terms can be shown equal, they will stay equal even if an undefined term will be identified with a defined term later on. On the other hand might an undefined term become equal (w. r. t. $\overset{\oplus}{\Longleftrightarrow}$) to a previously unequal term when identifying an undefined term with a defined term.

[1] i. e.: all terms in such a "constructor rule" are constructor terms and all variables of a constructor rule occur in its left-hand side

The changes of (A),(B) have the following advantages (compared to Kaplan[6]) (cf. sect. 5):

1. $\overset{\oplus}{\Longleftrightarrow}$ yields a minimal model that is (beyond that) the minimum of all models of R that do not identify more defined terms than necessary.
2. $\Longrightarrow$ is monotonic w. r. t. the addition of new rules that don't have left-hand sides that are old constructor terms.
3. $\Longrightarrow$ is stable when defined also on non-ground terms

If there aren't any undefined ground terms, then there is no difference from the reduction relation of Kaplan[6]: In this important case we offer semantics for [6] not using any control information.

Finally, we are not only able to specify our semantics without using control information, but also able to remove the control aspect of requiring the rules to be decreasing. As a consequence, our reduction relation does not need to be noetherian or restricted to ground terms.

4 Syntax and Semantics of the Specification Language

Definition 4.1 (Syntax of CRS) *A (p/n-)*conditional rule system *(*CRS*)* R *over* sig, cons *and* V *is a finite subset of the* set of rules $\mathcal{RUL}(\mathrm{sig},\mathrm{cons})$ *over* sig, cons *and* V*, that will be defined in Definition 5.1. The only thing we have to know about it now is:* $\mathcal{RUL}(\mathrm{sig},\mathrm{cons}) \subseteq \mathrm{DEq}(\mathrm{sig}) \times (\mathcal{LIT}(\mathrm{sig}))^*$, *where* DEq(sig) *is the set of* directed equations *and* $\mathcal{LIT}(\mathrm{sig})$ *is the set of* condition literals *over the meta-predicate*[2] *symbols on terms '$=$', '$\neq$' (binary, commutative), and '*Def*' (unary). The set of* condition terms *of a rule* $((l,r),C)$ *is denoted by* $\mathcal{TERMS}(C)$*.* $((l,r),C)$ *is said to be* extra-variable free *:iff* $\mathcal{V}(\mathcal{TERMS}(C),r) \subseteq \mathcal{V}(l)$*.* R *is* extra-variable free *:iff all its rules are extra-variable free.*

A rule $((l,r),C)$ from $\mathcal{RUL}(\mathrm{sig},\mathrm{cons})$ expresses a universally quantified implication with the conjunction of the literals in C as the condition and with '$l{=}r$' as the conclusion. The meaning of the meta-predicate symbols '$=$', '$\neq$', and 'Def' is not open to interpretation: '$=$' is standard; '$\neq$' is its negation; 'Def', however, is the "definedness" predicate which states that the evaluation of its argument belongs (with sort invariant) to the cons-term-generated cons-subalgebra of $\mathcal{A}$. We speak of our new kind of model just as a "sig/cons-model", because it is an upward-compatible extension (for 'Def') of the usual model concept of algebra.

[2] It is also possible to put 'Def' into the signature 'sig' as a predicate symbol whose semantics must then be given by each sig/cons-algebra $\mathcal{A}$ as a subfamily of $(\mathcal{A}[\mathcal{GT}(\mathrm{cons})_s])_{s\in S}$. E. g. for finitely generated categories the composition of two morphisms should only be "defined" if the codomain of the first is equal to the domain of the second:

$$\mathrm{Def}(f\circ g) \;\longleftarrow\; \mathrm{Def}\, f,\ \mathrm{Def}\, g,\ \mathrm{Cod}(f){=}\mathrm{Dom}(g)$$

Since this more flexible approach requires a lot of additional terminology and basic theory, we choose the simpler approach due to lack of space here.

Definition 4.2 (Semantics of CRS) *Let* R *be a CRS over* sig, cons *and* V*; let* $\mathcal{A}$ *be a* sig*-algebra. Now* $\mathcal{A}$ *is a* sig/cons-model of R <u>:iff</u>
$\forall((l,r),C)\in \mathrm{R} : \forall b$ $\mathcal{A}$*-valuation of* V :

$$((C \textit{ is } \textbf{true w. r. t. } \mathcal{A}_b) \;\Rightarrow\; \mathcal{A}_b(l)=\mathcal{A}_b(r))\ ,$$

where C *is* **true w. r. t.** $\mathcal{A}_b$ <u>:iff</u>

$$\forall s\in \mathrm{S} : \forall u,v\in \mathcal{T}(\mathrm{sig})_s : \begin{pmatrix} ((\ (u{=}v\ )\ \textit{in } C) \;\Rightarrow\; \mathcal{A}_b(u)=\mathcal{A}_b(v)\ ) & \wedge \\ ((\ (u{\neq}v\ )\ \textit{in } C) \;\Rightarrow\; \mathcal{A}_b(u)\neq\mathcal{A}_b(v)\ ) & \wedge \\ ((\ (\mathrm{Def}\, u)\ \textit{in } C) \;\Rightarrow\; \mathcal{A}_b(u)\in\mathcal{A}[\mathcal{GT}(\mathrm{cons})_s]\ ) & \end{pmatrix}$$

As we have negative equations in our conditions, we cannot hope to get a minimum model because we can express things like '$a=b \vee b=c$', which has the incomparable minimal models '$a=b\neq c$' and '$a\neq b=c$'. What we will get instead is a model that is the minimum of all models that do not identify more defined terms than necessary. For formally expressing this minimality-property, we need the following definition.

Definition 4.3 *Define the quasi-orderings* $\lesssim_{\mathrm{H}}$ *and* $\lesssim_{\mathrm{cons}}$ *as (proper class) relations on* sig*-algebras by:* $\mathcal{A}\lesssim_{\mathrm{H}}\mathcal{B}$ **:iff** *there is a* sig*-homomorphism from* $\mathcal{A}$ *to* $\mathcal{B}$*.* $\mathcal{A}\lesssim_{\mathrm{cons}}\mathcal{B}$ **:iff** *there is a* cons*-homomorphism from the* cons*-subalgebra of* $\mathcal{A}$ *with the universes* $\mathcal{A}[\mathcal{GT}(\mathrm{cons})_s]$ *to the analogous subalgebra of* $\mathcal{B}$*. We trivially get* $\lesssim_{\mathrm{H}}\subseteq\lesssim_{\mathrm{cons}}$*. The corresponding equivalences, orderings, and reflexive orderings will be denoted by* $\approx$*,* $<$*,* $\leq$*, resp., with the corresponding subscript. A* sig*-algebra* $\mathcal{A}$ *will be called a* minimum model *(or else a* constructor-minimum model*) of a CRS* R *over* sig/cons/V **:iff** $\mathcal{A}$ *is a* $\lesssim_{\mathrm{H}}$*-minimum (or else* $\lesssim_{\mathrm{cons}}$*) of the class of all* sig/cons*-models of* R*. Similarly, a* sig*-algebra* $\mathcal{A}$ *will be called a* minimal model *(or else a* constructor-minimal model*) of a CRS* R *over* sig/cons/V **:iff** $\mathcal{A}$ *is a* sig/cons*-model of* R *and there is no* sig/cons*-model* $\mathcal{B}$ *of* R *with* $\mathcal{B}<_{\mathrm{H}}\mathcal{A}$ *(or else* $\mathcal{B}<_{\mathrm{cons}}\mathcal{A}$ *).*

The following lemma tells us that, considering minimum models, we can think in terms of congruences on $\mathcal{GT}(\mathrm{sig})$ instead of algebras:

Lemma 4.4 *Let* $\mathcal{B}$ *be a* sig/cons*-model of the CRS* R *over* sig/cons/V*. Define* $\sim$ *by* ($s,t\in\mathcal{GT}(\mathrm{sig})$)*:* $s\sim t$ <u>:iff</u> $\mathcal{B}(s)=\mathcal{B}(t)$*. Define the factor algebra* $\mathcal{A}:=\mathcal{GT}(\mathrm{sig})/\!\sim$ *. Now:*

1. $\mathcal{A}$ *is a* sig/cons*-model of* R*.*

2. $\mathcal{A}\lesssim_{\mathrm{H}}\mathcal{B}$ *. Moreover, there is a unique (injective)* sig*-homomorphism from* $\mathcal{A}$ *to* $\mathcal{B}$*.*

3. $\mathcal{A}\approx_{\mathrm{cons}}\mathcal{B}$

Our sig/cons-model concept is algebraically well-behaved in that the trivial algebra is always a sig/cons-model and for each model there is a smaller one that is minimal (cf. [9]).

5 The Reduction Relation

Many authors impose rather strong restrictions on constructor equations, such as "no equations between constructors" ("free constructors") or "unconditional equations between constructors only". Compared to these, our restrictions are very weak. They serve to guarantee a constructor-minimum model for the constructor equations that is unique modulo $\approx_{\rm cons}$, by requiring the constructor equations to be "constructor-preserving"[3] and to have "Horn"-form.

Definition 5.1 (Set of Rules)
(continuing Definition 4.1 by adding the restrictions on constructor equations)
The set of rules *over* sig, cons *and* V *is defined to be:* $\mathcal{RUL}(\mathrm{sig},\mathrm{cons}) := \{((l,r),C) \in (\mathrm{DEq}(\mathrm{sig}) \times (\mathcal{LIT}(\mathrm{sig}))^*) \mid (\ l{\in}\mathcal{T}(\mathrm{cons}) \ \Rightarrow\ (\mathcal{V}(r,\mathcal{TERMS}(C)) \subseteq \mathcal{V}(l) \ \wedge\ r{\in}\mathcal{T}(\mathrm{cons}) \ \wedge\ \mathcal{TERMS}(C) \subseteq \mathcal{T}(\mathrm{cons}) \ \wedge\ \forall L \textit{ in } C\colon \forall u,v\colon\ L \neq (u{\neq}v)))\}$

We are now going to define our reduction relation, having in mind to require it to be confluent in the sequel, whereas we do not require confluence for the definition because we cannot prove confluence criteria if the non-confluent case is undefined. Therefore, we have to be explicit how we test the condition literals — even if this testing is not straightforward when confluence is not provided. Our "operational" semantics for testing condition literals is the following: '$u{=}v$' is true if u,v have reducts $\hat{u},\hat{v}$, resp., which are syntactically equal. 'Def u' is true if u has a constructor ground reduct. '$u \neq v$' is true if u,v have constructor ground reducts $\hat{u},\hat{v}$, resp., which are not joinable. Thus, two terms in a condition literal are "operationally" equal if they are joinable, whereas they are unequal if they are not joinable after some reduction to constructor ground terms. The non-joinability alone of two terms is not sufficient for regarding them as unequal because we are never sure about the inequality of "undefined" terms. As it often occurs, our operational logic is four-valued (i. e. $=$ and $\neq$ can independently be true or false), but in case of confluence: *tertium non datur*. In case of free or confluent constructors, the case of both '$u{=}v$' and '$u{\neq}v$' simultaneously being true means that we have something like an ambiguous function definition.

---

[3]The constructor-preservation is really necessary here for guaranteeing a constructor-minimum model as in Theorem 5.14: Let $\mathsf{0}, \mathsf{1}, \mathsf{true}, \mathsf{false}$ be constructor constants, let weirdp be a defined constant, and take R:

$$\begin{array}{lcl} \mathsf{1} = \mathsf{0} & \longleftarrow & \mathsf{weirdp} = \mathsf{true} \quad ; \\ \mathsf{weirdp} = \mathsf{true} & \longleftarrow & \mathsf{true} \neq \mathsf{false} \quad . \end{array}$$

Now there are models of R with $\mathsf{0} \neq \mathsf{1}$ and models with $\mathsf{true} \neq \mathsf{false}$ but no models with $\mathsf{0} \neq \mathsf{1}$ and $\mathsf{true} \neq \mathsf{false}$. Also notice, that the constructor-preservation has some additional advantages, e. g.:

1. For $u \in \mathcal{GT}(\mathrm{sig})$ and computable and unique normal form $\mathrm{NF}(u)$ we can test " $\exists \hat{u} \in \mathcal{GT}(\mathrm{cons}) : u \overset{\circledast}{\Longrightarrow} \hat{u}$ " by " $\mathrm{NF}(u) \in \mathcal{GT}(\mathrm{cons})$ ".
2. Theorem 5.15 has no reasonable analogue for CRSs which are not constructor-preserving.

Another important aspect of our reduction relation is that it depends on the constructor subsignature 'cons' beyond the signature 'sig' — just as our notion of a "sig/cons-model" does.

Definition 5.2 (Our Reduction Relation $\Longrightarrow_{\mathrm{R}}$)
Let R *be a CRS over* sig/cons/V*. The reduction relation* $\Longrightarrow_{\mathrm{R}}$ *on* $\mathcal{T}(\mathrm{sig})$ *(*$\Longrightarrow$ *for short) is defined to be the smallest relation satisfying the following requirement (:#):* $s \Longrightarrow t$ $\underline{\text{if}}$

$$\exists((l,r),C)\in\mathrm{R}\colon \exists\sigma\in\mathcal{SUB}(\mathrm{V},\mathcal{T}(\mathrm{sig}))\colon \exists p\in\mathcal{O}(s)\colon \begin{pmatrix} s/p & = & l\sigma & \wedge \\ t & = & s[\,p\leftarrow r\sigma\,] & \wedge \\ C\sigma & \multicolumn{2}{l}{\textit{is } \textbf{fulfilled w. r. t.} \Longrightarrow} \end{pmatrix},$$

where "C is **fulfilled w. r. t.** $\Longrightarrow$*" is a shorthand for* $\forall u,v\in\mathcal{T}(\mathrm{sig})$:

$$\begin{pmatrix} (\ (\ (u{=}v\ )\ in\ C) & \Rightarrow & & u\downarrow v\ \ ) & \wedge \\ (\ (\ (\mathrm{Def}\,u)\ in\ C) & \Rightarrow & \exists\hat{u}\in\mathcal{GT}(\mathrm{cons}): & u\overset{\oplus}{\Longrightarrow}\hat{u}\ ) & \wedge \\ (\ (\ (u{\neq}v\ )\ in\ C) & \Rightarrow & \exists\hat{u},\hat{v}\in\mathcal{GT}(\mathrm{cons}): & u\overset{\oplus}{\Longrightarrow}\hat{u}\not\downarrow\hat{v}\overset{\oplus}{\Longleftarrow}v\ )^{4} & \end{pmatrix}$$

Usually one tries to find a minimal reduction relation by taking the closure over a finitary generating relation. This is not possible here, because we have a negative condition ($\not\downarrow$). By the "Horn"-form of our constructor equations (and the constructor-preservation), however, we know that this negative condition does not influence the reduction of constructor terms; and (in the definition) '$\not\downarrow$' is applied to constructor (ground) terms only. Thus, we can get our minimal reduction relation by a double closure: first for constructor terms only; second for general terms knowing the constructor reduction to remain unchanged. This two step construction does not destroy the uniformity of the defining requirement(#), which allows us to write uniform normal form procedures, provided that the normal forms are computable.

Define $\Longrightarrow_{\mathrm{R},0} := \emptyset$ and $\Longrightarrow_{\mathrm{R},i+1}$ to be the left-hand side[5] of the requirement(#) of Definition 5.2 with $\Longrightarrow_{\mathrm{R},i}$ substituted for $\Longrightarrow$ on the right-hand side and the additional restriction of $s\in\mathcal{T}(\mathrm{cons})$; formally: $s\Longrightarrow_{\mathrm{R},i+1}t$ $\underline{\text{:iff}}$ $s\in\mathcal{T}(\mathrm{cons})\ \wedge\ \exists((l,r),C)\in\mathrm{R}\colon\exists\sigma\in\mathcal{SUB}(\mathrm{V},\mathcal{T}(\mathrm{sig}))\colon\exists p\in\mathcal{O}(s)\colon(s/p = l\sigma\ \wedge\ t = s[\,p\leftarrow r\sigma\,]\ \wedge\ C\sigma$ is **fulfilled w. r. t.** $\Longrightarrow_{\mathrm{R},i})$.

[4] This formulation requires confluence and constructor-preservation to make sense in two-valued logic. While other formulations (e. g. a universal instead of the existential quantification) might seem more satisfactory, this is the one required for a correct definition.

One might have expected $u\not\downarrow v$ instead of $\exists\hat{u},\hat{v}\in\mathcal{GT}(\mathrm{cons}):\ u\overset{\oplus}{\Longrightarrow}\hat{u}\not\downarrow\hat{v}\overset{\oplus}{\Longleftarrow}v$ for "$u{\neq}v$ in C" here, but this modification would not allow us to conclude that $\Longrightarrow_{\mathrm{R}}$ is uniquely defined as can be seen from the following example:

Example 5.3
Let a,b,c,d,e *be constants and take* R*:*

$$\begin{array}{lll} \mathsf{a} = \mathsf{c} \longleftarrow \mathsf{b}\neq\mathsf{d} & ; \\ \mathsf{b} = \mathsf{d} \longleftarrow \mathsf{e}\neq\mathsf{c} & ; \\ \mathsf{e} = \mathsf{a} & . \end{array}$$

The intersection of { (a, c), (e, a) } *and* { (b, d), (e, a) } *(which would be minimal relations satisfying the modified requirement(#) of Definition 5.2) would not satisfy the modified requirement of Definition 5.2 anymore.*

[5] the ' $\underline{\text{if}}$ ' replaced by ' $\underline{\text{:iff}}$ '

Define $\Longrightarrow_{\mathrm{R},\omega} := \bigcup_{i\in\mathbb{N}} \Longrightarrow_{\mathrm{R},i}$ and $\Longrightarrow_{\mathrm{R},\omega+i+1}$ to be the union of $\Longrightarrow_{\mathrm{R},\omega}$ and the left-hand side[5] of the requirement(#) of Definition 5.2 with $\Longrightarrow_{\mathrm{R},\omega+i}$ substituted for $\Longrightarrow$ on the right-hand side. Finally, define $\Longrightarrow_{\mathrm{R}} := \Longrightarrow_{\mathrm{R},\omega+\omega} := \bigcup_{i\in\mathbb{N}} \Longrightarrow_{\mathrm{R},\omega+i}$. Now $\Longrightarrow_{\mathrm{R}}$ satisfies the requirement(#) of Definition 5.2, and every relation satisfying this requirement must contain $\Longrightarrow_{\mathrm{R}}$. Hence $\Longrightarrow_{\mathrm{R}}$ is the intended smallest relation.

By double induction over the above construction process it is trivial to verify the following corollaries.

Corollary 5.4 (Monotony of $\Longrightarrow_{\mathrm{R}}$ w. r. t. Replacement)
$\Longrightarrow_{\mathrm{R},i}$, $\Longrightarrow_{\mathrm{R},\omega+i}$ (for $i\in\mathbb{N}$), $\Longrightarrow_{\mathrm{R}}$ are monotonic.

Corollary 5.5 (Stability of $\Longrightarrow_{\mathrm{R}}$) *$\Longrightarrow_{\mathrm{R},i}$, $\Longrightarrow_{\mathrm{R},\omega+i}$ (for $i\in\mathbb{N}$), $\Longrightarrow_{\mathrm{R}}$ and their respective fulfilledness-predicates are stable.*

Lemma 5.6 (Monotony of $\Longrightarrow_{\mathrm{R},\beta}$ in β) *For two ordinal numbers β,γ smaller than $\omega+\omega$, if β is smaller than γ (i. e. if $\beta \preceq \gamma$), then $\Longrightarrow_{\mathrm{R},\beta} \subseteq \Longrightarrow_{\mathrm{R},\gamma}$.*

The following technical lemmas state constructor-preservation and that there is no need for a second closure for reduction of constructor terms.

Lemma 5.7
$$\forall n\in\mathbb{N} : \forall s\in\mathcal{T}(\mathrm{cons}) : \forall t : \left(\ s \overset{n}{\Longrightarrow} t \quad\Rightarrow\quad (s \overset{n}{\Longrightarrow}_{\mathrm{R},\omega} t \in \mathcal{T}(\mathrm{cons}))\ \right)$$

Lemma 5.8
$$\forall n\in\mathbb{N} : \forall s\in\mathcal{GT}(\mathrm{cons}) : \forall t : \left(\ s \overset{n}{\Longrightarrow} t \quad\Rightarrow\quad (s \overset{n}{\Longrightarrow}_{\mathrm{R},\omega} t \in \mathcal{GT}(\mathrm{cons}))\ \right)$$

Lemma 5.9 $\quad \downarrow \cap (\mathcal{T}(\mathrm{cons})\times\mathcal{T}(\mathrm{cons})) \quad = \quad \downarrow_{\mathrm{R},\omega}$

Lemma 5.10 (Fulfilledness Test may be Simple)
Let $C\in\mathcal{LIT}(\mathrm{sig})^*$. If for each element $u\in\mathcal{TERMS}(C)$:

1. *u has a [ground] normal form $\mathrm{NF}(u)$*
   (i. e. $u \overset{\circledast}{\Longrightarrow} \mathrm{NF}(u) \notin \mathrm{dom}(\Longrightarrow)$[, $\mathrm{NF}(u)\in\mathcal{GT}(\mathrm{sig})$])

and

2. *$\Longrightarrow$ is [ground] confluent below u [and $\mathcal{V}(\mathcal{TERMS}(C))=\emptyset$],*

then C is fulfilled w. r. t. $\Longrightarrow$ <u>iff</u>
$\forall u,v\in\mathcal{T}(\mathrm{sig})$:
$$\begin{pmatrix} (\ (\ (u{=}v\ )\ \mathit{in}\ C) & \Rightarrow & \mathrm{NF}(u)=\mathrm{NF}(v)\ \ ) & \wedge \\ (\ (\ (\mathrm{Def}\, u)\ \mathit{in}\ C) & \Rightarrow & \mathrm{NF}(u)\in\mathcal{GT}(\mathrm{cons})\ ) & \wedge \\ (\ (\ (u{\neq}v\ )\ \mathit{in}\ C) & \Rightarrow & (\mathrm{NF}(u),\mathrm{NF}(v)\in\mathcal{GT}(\mathrm{cons})\ \wedge\ \mathrm{NF}(u)\neq\mathrm{NF}(v))\ ) & \end{pmatrix}$$

Considering ground confluence, it is important to realize that for reduction on ground terms we can restrict ourselves to ground substitutions:

Lemma 5.11
(Ground Substitutions are Sufficient for Ground Reduction)
Let $\Longrightarrow_{\mathrm{R,ground}}$ *be defined just like* $\Longrightarrow_{\mathrm{R}}$ *with the exception of the additional restriction "*$\sigma \in \mathcal{SUB}(\mathrm{V}, \mathcal{GT}(\mathrm{sig})) \ \wedge \ s \in \mathcal{GT}(\mathrm{sig})$*" in the requirement(#) of Definition 5.2.*

Now $\Longrightarrow_{\mathrm{R,ground}}$ *is uniquely defined,* $\Longrightarrow_{\mathrm{R,ground}} = \Longrightarrow_{\mathrm{R}} \cap (\mathcal{GT}(\mathrm{sig}) \times \mathcal{GT}(\mathrm{sig}))$, $\overset{\oplus}{\Longrightarrow}_{\mathrm{R,ground}} = \overset{\oplus}{\Longrightarrow}_{\mathrm{R}} \cap (\mathcal{GT}(\mathrm{sig}) \times \mathcal{GT}(\mathrm{sig}))$*, and* $\mathrm{dom}(\Longrightarrow_{\mathrm{R,ground}}) = \mathrm{dom}(\Longrightarrow_{\mathrm{R}}) \cap \mathcal{GT}(\mathrm{sig})$*. Furthermore,* $\Longrightarrow_{\mathrm{R,ground}}$ *is confluent* *iff* $\Longrightarrow_{\mathrm{R}}$ *is ground confluent.*

By Example 5.3 we know that (for guaranteeing a constructor-minimum model) we have to restrict the terms of the negated equations to be "defined". This semantic restriction is made syntactically explicit in the following definition that specifies a "well-behaved" subclass of the class of CRSs, in which inequalities are founded on constructor ground terms.

Definition 5.12 (Moderate Conditional Rule Systems (MCRS))
A CRS R *is a* moderate conditional rule system *(*MCRS*)* *:iff*

$$\forall((l,r),C) \in \mathrm{R} : \forall(u{\neq}v) \ \textit{in} \ C : \big(\ \mathrm{Def}\, u, \mathrm{Def}\, v \ \textit{are in} \ C \ \big) \ .$$

For a motivation of MCRSs see item (A) in section 3, where we discussed the problems involved.

The following definition allows us to use a CRS as a shorthand for a MCRS:

Definition 5.13 (Moderated Version of a CRS)
The moderated version of a CRS R *is the MCRS* R′ *obtained by replacing each sequence of the form '*$u{\neq}v$*' in each condition of each rule in* R *by the*[6] *sequence '*$u{\neq}v$, $\mathrm{Def}\, u$, $\mathrm{Def}\, v$*'.*

Now we are able to state the fundamental theorem about $\Longrightarrow$. It says that for moderate CRSs R with (ground) confluent $\Longrightarrow_{\mathrm{R}}$, the factor algebra $\mathcal{GT}(\mathrm{sig})/\overset{\oplus}{\Longleftrightarrow}_{\mathrm{R}}$ is an (up to isomorphism) uniquely determined sig/cons-model of R.

[6] We think the symmetry in u and v to be removed in some deterministic manner

Theorem 5.14 (Minimal Model being the Minimum of the Constructor-Minimal Models)

Let R *be a MCRS over* $\mathrm{sig/cons/V}$.
Furthermore, assume $\Longrightarrow_{\mathrm{R}}$ *to be ground confluent.*

1. $\mathcal{GT}(\mathrm{sig})/\overset{\oplus}{\Longleftrightarrow}$ *is a constructor-minimum*[7] *model of* R.

2. *Let* $\mathcal{B}$ *be a model of* R *with*

   (a) $\mathcal{B}$ *a constructor-minimal*[7] *model of* R
   or
   (b) $\mathcal{B} \precsim_{\mathrm{H}} \mathcal{GT}(\mathrm{sig})/\overset{\oplus}{\Longleftrightarrow}$.

   Now $\mathcal{GT}(\mathrm{sig})/\overset{\oplus}{\Longleftrightarrow} \precsim_{\mathrm{H}} \mathcal{B}$.

   In other words: $\mathcal{GT}(\mathrm{sig})/\overset{\oplus}{\Longleftrightarrow}$ *is (by 2b) a minimal model and (by 2a) the minimum of all models that do not identify more defined terms than necessary.*

Theorem 5.15
(Monotony of $\Longrightarrow_{\mathrm{R}}$ **w. r. t. Consistent Extension of the Specification)**

Let R *be a CRS over* sig, cons *and* V. *Let* R' *be another CRS, but over* sig', cons' *and* V'*; with*

$$\left|\begin{array}{l} \mathrm{sig}' = (\mathrm{F}', \mathrm{S}', \alpha') \\ \mathrm{cons}' = (\mathrm{C}', \mathrm{S}', \alpha'|_{\mathrm{C}'}) \\ \mathrm{V}'|_{\mathrm{S}} = \mathrm{V} \end{array}\right| \begin{array}{l} \mathrm{F} \subseteq \mathrm{F}' \\ \mathrm{C} \subseteq \mathrm{C}' \subseteq \mathrm{F}' \\ \mathrm{S} \subseteq \mathrm{S}' \\ \alpha \subseteq \alpha' \end{array} \left| \ \mathrm{R} \subseteq \mathrm{R}' \ \right|$$

Thus, $\mathrm{sig}'/\mathrm{cons}'/\mathrm{V}'$ *is an enrichment of* $\mathrm{sig/cons/V}$ *in the most general*[8] *sense we can think of.*

Moreover, assume[9]*:* $\forall((l,r),C) \in (\mathrm{R}' \setminus \mathrm{R}) : l \notin \mathcal{T}(\mathrm{cons})$ *(:$)*

Now we have:

1. $\forall s \in \mathcal{T}(\mathrm{cons}) : \forall t : \Big((s \overset{\oplus}{\Longrightarrow}_{\mathrm{R}} t) \ \Leftrightarrow \ (s \overset{\oplus}{\Longrightarrow}_{\mathrm{R}'} t) \Big)$
   "no change on old constructor terms"

2. $\Longrightarrow_{\mathrm{R}} \ \subseteq \ \Longrightarrow_{\mathrm{R}'}$ *"monotony"*

[7] Cf. Definition 4.3
[8] One may even define new constructor function symbols for the old sorts and take them from the old defined symbols.
[9] This has to be required for keeping the negative conditions fulfilled: Having founded our inequalities on old constructor ground terms, all we have to take care of now is not to confuse these terms.

6 Confluence

We are now going to define critical peaks that consist of the conditional critical pair, its peak, and the overlap position. The other notions we will use are also standard, with the exception of the notion of "quasi overlay joinable" which is a slight weakening (allowing a constant non-overlay part) of "joinable overlays" in Dershowitz[3].

Definition 6.1 *The set of (non-trivial)* critical peaks *between two rules* $((l_k, r_k), C_k) \in \mathrm{R}$ *(whose common variables we assume to be renamed) is defined by:* $\mathrm{Cp}(\ ((l_0, r_0), C_0),\ ((l_1, r_1), C_1)\ ) := \{(((l_1[p \leftarrow r_0], r_1), C_0 C_1)\sigma,\ \ l_1\sigma,\ \ p) \mid p \in \mathcal{O}(l_1) \wedge l_1/p \notin \mathrm{V} \wedge (\sigma \text{ most general unifier for } l_0, l_1/p) \wedge l_1[p \leftarrow r_0]\sigma \neq r_1\sigma\}$. *The set of all critical peaks of* R *is* $\mathrm{CP(R)} := \bigcup_{\mathrm{rule}_0 \in \mathrm{R}} \bigcup_{\mathrm{rule}_1 \in \mathrm{R}} \mathrm{Cp}(\mathrm{rule}_0, \mathrm{rule}_1)$. R *is said to be* overlapping *:iff* $\mathrm{CP(R)} \neq \emptyset$. *A critical peak* $(((t_0, t_1), D), \hat{t}, p)$ *is* weakly [ground] joinable *(w. r. t.* $\Longrightarrow$*)* *:iff* $\forall \varphi \in \mathcal{SUB}(\mathrm{V}, \mathcal{T}(\mathrm{sig})[\cap \mathcal{GT}(\mathrm{sig})]) : ((D\varphi \text{ fulfilled} \wedge \forall u \in \mathcal{TERMS}(D\varphi) : \Longrightarrow \text{ is confluent below } u) \Rightarrow t_0\varphi \downarrow t_1\varphi)$. *A critical peak* $(((t_0, t_1), D), \hat{t}, p)$ *is* quasi overlay [ground] joinable *(w. r. t.* $\Longrightarrow$*)* *:iff* $\forall \varphi \in \mathcal{SUB}(\mathrm{V}, \mathcal{T}(\mathrm{sig})[\cap \mathcal{GT}(\mathrm{sig})])$:

$$\left(D\varphi \text{ fulfilled } \Rightarrow \left(\begin{array}{l} t_1\varphi = t_0\varphi[p \leftarrow t_1\varphi/p] \quad \wedge \\ (t_0/p)\varphi \downarrow t_1\varphi/p \overset{\oplus}{\Longleftarrow} (\hat{t}/p)\varphi \end{array} \right) \right) .$$

The proof of the following theorem is similar to that of Theorem 4 in Dershowitz[3]:

Theorem 6.2 (Confluence Criterion) *Let* R *be a CRS over* sig*,* cons *and* V*. If* $\Longrightarrow_{\mathrm{R}}$ *is [ground]*[10] *noetherian and all critical peaks in* CP(R) *are quasi overlay [ground] joinable, then* $\Longrightarrow_{\mathrm{R}}$ *is [ground] confluent.*

In the sequel we assume two fixed stable orderings $>$ and $\rhd$ on terms, such that $>$ is monotonic and $\rhd$ is well-founded and contains $>$ and the proper subterm ordering. We call a rule $((l, r), C) \in \mathcal{RUL}(\mathrm{sig}, \mathrm{cons})$ *aligned* (w. r. t. $(>, \rhd)$) :iff

$$l > r \ \wedge \ \forall u \in \mathcal{TERMS}(C) : \ l \rhd u \ .$$

We have proper reason for this form (cf. [9]) but not the space to explain it here.

The following kind of compatibility is a generalization to negative conditions and a slight weakening of the notion of "decreasingness" in Dershowitz[3].

Definition 6.3 ([Ground] Compatibility of a CRS)
A CRS R *over* sig/cons/V *is* [ground] compatible *:iff*
$\forall ((l, r), C) \in \mathrm{R} : \forall \tau \in \mathcal{SUB}(\mathrm{V}, \mathcal{T}(\mathrm{sig})[\cap \mathcal{GT}(\mathrm{sig})]) :$

$$(C\tau \text{ fulfilled} \Rightarrow (\ ((l, r), C)\tau \text{ is aligned }\ ))$$

[10] doesn't make a difference of course (cf. Corollary 5.5)

The [ground] compatibility of a CRS R guarantees the alignment of a [ground] instantiated rule of R when its condition is fulfilled. But, while this kind of compatibility is convenient for obtaining further theoretical properties of the reduction relation, we have a problem when using this kind of compatibility of R in practice of reduction: The terms in $C\tau$ must be smaller than $l\tau$ only if $C\tau$ is fulfilled; but for easily deciding whether $C\tau$ is fulfilled we need its terms to be smaller than $l\tau$ and the analogous property for the other rules. That this need not be a vicious circle is shown by the following definition, which allows us to test the literals in the condition from left to right, using the old programming trick of a sequential "short-circuiting" AND-operator[11]. Notice that the difference to Definition 6.3 is in the quantified variable i occurring as an index which allows us to step inductively from $(< i)$ to i.

Definition 6.4 ([Ground] Left-Right-Compatibility)
A CRS R *over* sig/cons/V *is* [ground] left-right-compatible $\underline{:\!\mathit{iff}}$
$$\forall((l,r),L_0\ldots L_{n-1})\in \mathrm{R}:\forall\tau\in\mathcal{SUB}(\mathrm{V},\mathcal{T}(\mathrm{sig})[\cap\mathcal{GT}(\mathrm{sig})]):$$
$$\left(\begin{array}{lll} \forall i<n:(\ (L_0\ldots L_{i-1})\tau\ \mathit{fulfilled} & \Rightarrow \forall u\in\mathcal{TERMS}(L_i): l\tau\ \rhd u\tau\ ) \\ \wedge \qquad\quad (\ (L_0\ldots L_{n-1})\tau\ \mathit{fulfilled} & \Rightarrow l\tau > r\tau \qquad\qquad\qquad\quad) \end{array}\right)$$

Definition 6.5 ([Ground] Don't-Care-Compatibility)
A CRS R *over* sig/cons/V *is* [ground] don't-care-compatible $\underline{:\!\mathit{iff}}$
$$\forall((l,r),L_0\ldots L_{n-1})\in \mathrm{R}:\forall\tau\in\mathcal{SUB}(\mathrm{V},\mathcal{T}(\mathrm{sig})[\cap\mathcal{GT}(\mathrm{sig})]):$$
$$\left(\begin{array}{lll} \forall i<n: & & \forall u\in\mathcal{TERMS}(L_i): l\tau\ \rhd u\tau \\ \wedge \qquad\quad (\ (L_0\ldots L_{n-1})\tau\ \mathit{fulfilled} & \Rightarrow & l\tau > r\tau \qquad\qquad\qquad\quad) \end{array}\right)$$

Having a left-right-compatible CRS, we don't have to test the instantiated rules for alignment anymore, provided that we test the literals of the instantiated conditions from left to right until one of them fails. Having a don't-care-compatible CRS, we can even test the literals of the instantiated conditions in parallel and don't have to care for the position of these equations in the condition list. The don't-care-compatibility is conceptionally the same as the "decreasingness" in Dershowitz[3].

The "compatibility" of Definition 6.3 (which seems to be the least restrictive one tractable in theory[12]) is intended to be an interface for the generation of logically stronger kinds of "compatibility" that are useful in practice (cf. definitions 6.4 and 6.5), where the don't-care-compatibility of Definition 6.5 seems to be the most important one. For restrictions of $\Longrightarrow_{\mathrm{R}}$, however, even weaker kinds of "compatibility" than the one of Definition 6.3 may be sufficient.

For compatible CRSs we can now give a complete critical peak test à la Knuth-Bendix:

Theorem 6.6 (Confluence Test)
Let R *be [ground] compatible a CRS over* sig/cons/V*. Now:*
$\Longrightarrow_{\mathrm{R}}$ *is [ground] confluent* $\underline{\mathit{iff}}$ *all critical peaks in* CP(R) *are weakly [ground] joinable.*

[11] '&&' (instead of '&') in C. 'AND' in LISP.

[12] allowing a complete confluence test and appropriate inference systems for inductive theorem proving using the same orderings

7 Decidability of Reducibility

While reducibility of ground terms is in general neither semi- nor co-semi-decidable for non-overlapping, extra-variable free MCRSs with noetherian and confluent reduction relation, we can generalize Theorem 3.4 in Kaplan[5] in various directions:

Lemma 7.1 *Let* R *be a CRS over* sig/cons/V*.*

1. $\Longrightarrow_{\mathrm{R}}$*[-ground]-reducibility*[13] *is co-semi-decidable if a [ground] normal form for each [ground] term is computable*
(i. e. there is a computable (partial) function f *with* $\mathrm{dom}(f) =$
$\{s \in \mathcal{T}(\mathrm{sig})[\cap \mathcal{GT}(\mathrm{sig})] \mid \exists t \in \mathcal{T}(\mathrm{sig})[\cap \mathcal{GT}(\mathrm{sig})] : (s \overset{\oplus}{\Longrightarrow}_{\mathrm{R}} t \wedge t \notin \mathrm{dom}(\Longrightarrow_{\mathrm{R}}))\}$
such that $\forall s \in \mathrm{dom}(f) : s \overset{\oplus}{\Longrightarrow}_{\mathrm{R}} f(s) \in (\mathcal{T}(\mathrm{sig})[\cap \mathcal{GT}(\mathrm{sig})]) \setminus \mathrm{dom}(\Longrightarrow_{\mathrm{R}})$*).*
2. *A [ground] normal form for each [ground] term is computable (see above) if* $\Longrightarrow_{\mathrm{R}}$*[-ground]-reducibility is co-semi-decidable and*
$\forall s \in \mathcal{GT}(\mathrm{cons}) : \exists t : s \overset{\oplus}{\Longrightarrow}_{\mathrm{R}} t \notin \mathrm{dom}(\Longrightarrow_{\mathrm{R}})$.

Corollary 7.2 *Let* R *be a CRS over* sig/cons/V*. Assume* $\Longrightarrow_{\mathrm{R}}$ *to be noetherian. Now, co-semi-decidability of* $\Longrightarrow_{\mathrm{R}}$*[-ground]-reducibility is logically equivalent to computability of a [ground] normal form for each [ground] term.*

The usefulness of our kinds of compatibility is shown (w. r. t. Corollary 7.2) by:

Lemma 7.3 *Let* R *be a [ground] compatible CRS over* sig/cons/V*.*
Now we have $(\Longrightarrow_{\mathrm{R}}[\cap(\mathcal{GT}(\mathrm{sig}) \times \mathcal{GT}(\mathrm{sig}))]) \subseteq >$*, and* $\Longrightarrow_{\mathrm{R}}$ *is noetherian.*
Furthermore, if R *is extra-variable free and if*
(a) $\rhd[\cap(\mathcal{GT}(\mathrm{sig}) \times \mathcal{GT}(\mathrm{sig}))]$ *is decidable*
or
(b) R *is [ground] left-right-compatible,*
then $\Longrightarrow_{\mathrm{R}}$*[-ground]-reducibility is decidable and* $\{\ t \mid s \overset{\oplus}{\Longrightarrow}_{\mathrm{R}} t\ \}$ *is a finite computable set for all* $s \in \mathcal{T}(\mathrm{sig})[\cap \mathcal{GT}(\mathrm{sig})]$*.*

8 Conclusion

The presented constructor-based approach for positive/negative-conditional equational specifications provides the theoretical background for inductive theorem proving in such specifications as described in Wirth[8]. Various conceivable notions of inductive validity are presented and discussed there. Moreover, an abstract inference system in the style of [1] for proving inductive (constructor) validity of first-order clauses is given. Several refined versions of this inference system incorporating powerful contextual rewriting and simplification rules as well as a marking scheme are developed. In case of a (ground confluent, noetherian and) sufficiently complete base specification the notions of (ordinary) inductive validity and (inductive) constructor validity coincide.

[13] Following our notational conventions, *ground-reducibility* is reducibility w. r. t. $\Longrightarrow_{\mathrm{R,ground}}$ or reducibility of ground terms (cf. Lemma 5.11), which is different from inductive reducibility.

The idea of choosing a constructor-based approach for assigning adequate semantics to p/n-conditional equational specifications was heavily inspired by previous work of Kapur&Musser[7] (for the case of unconditional equations only) and Zhang[10] where problems with the usual notion of inductive (initial) validity, in particular concerning its non-monotonic behaviour w. r. t. consistent extensions, and partially defined functions are treated. Under the appropriate syntactical restrictions (mentioned above) concerning the form of p/n-conditional rules it turns out that the combination of these ideas with the approach of Kaplan[6] becomes very fruitful and also relevant for practical purposes since many natural specifications involve both conditional equations (rules) with positive and negative conditions and partially defined functions.

The perfect model semantics approach of Bachmair&Ganzinger[2], which also includes a completion procedure, generalizes Kaplan's approach [6] by abstracting the control information hidden in the syntactic form of the rules into a reduction ordering which must be total on ground terms and which determines the construction process of the perfect model. In our analysis, however, reduction orderings used for orienting p/n-systems do not necessarily have to be total on ground terms. Very recently, another method for inductive theorem proving of first-order clauses has been developed in [4] along the lines of [2] by means of a transformational approach. Its relation to our approach in theory and practice remains to be investigated.

Acknowledgement: We would like to thank Ulrich Kühler for many valuable discussions and detailed criticisms on earlier versions of this paper.

References

[1] Leo Bachmair. *Proof by Consistency in Equational Theories.* 3^{rd} LICS 1988, pp. 228-233.

[2] Leo Bachmair, Harald Ganzinger. *Perfect Model Semantics for Logic Programs with Equality.* Proc. of 8^{th} Int. Conf. on Logic Programming, pp. 645-659, MIT Press, 1991.

[3] Nachum Dershowitz, Mitsuhiro Okada, G. Sivakumar. *Confluence of Conditional Rewrite Systems.* LNCS 308. Springer-Verlag, Berlin 1988.

[4] Harald Ganzinger, Jürgen Stuber. *Inductive Theorem Proving by Consistency for First-Order Clauses.* This volume.

[5] Stéphane Kaplan. *Conditional Rewrite Rules.* Theoretical Computer Science 33 (1984), pp. 175-193. North-Holland.

[6] Stéphane Kaplan. *Positive/Negative-Conditional Rewriting.* LNCS 308. Springer-Verlag, Berlin 1988.

[7] Deepak Kapur, David R. Musser. *Proof by Consistency.* Artificial Intelligence 31 (1987), pp. 125-157.

[8] Claus-Peter Wirth. *Inductive Theorem Proving in Theories specified by Positive/Negative Conditional Equations.* Diplomarbeit, 1991, Universität Kaiserslautern, Fachbereich Informatik, Postfach 3049, W-6750 Kaiserslautern.

[9] Claus-Peter Wirth, Bernhard Gramlich. *A Constructor-Based Approach for Positive/Negative-Conditional Equational Specifications.* SEKI-Report SR-92-10 (SFB), May 4, 1992, Universität Kaiserslautern, Fachbereich Informatik.

[10] Hantao Zhang. *Reduction, Superposition and Induction: Automated Reasoning in an Equational Logic.* Rensselaer Polytech. Inst., Dept. of Comp. Sci., Troy, NY, PhD thesis, 1988.