Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern

SEKI - REPORT



# Inductive Proofs in Specifications Parametrized by a Built-in Theory

Klaus Becker

SEKI Report SR-92-02 (SFB)

# Inductive Proofs in Specifications Parametrized by a Built-in Theory

Klaus Becker
Fachbereich Informatik
Universität Kaiserslautern
6750 Kaiserslautern
Germany

## Abstract

We consider the problem of proving hypotheses to be valid in all canonical term models of some parametrized positive/negative conditional equational specification that are induced by the actualizations of the given parameter theory.

In a first part we develope a method to operationally handle as well the parameter actualizations as the body specification and here in particular the negative conditions occuring possibly in the conditional equations. By introducing restrictions on the conditional equations the induced canonical term algebras become in a certain sense initial models of the specification.

In a second part we present a proof by consistency method to deal with hypotheses formulated as clauses over the body and (formal) parameter language. If some syntactical premises are fulfilled and if a (built-in) algorithm is available that decides whether clauses over the parameter language only are logical consequences of the parameter theory or not, then the proof method becomes refutationally complete.

# 1 Introduction

Automation of proving theorems that are valid in some "particular intended" models of a specification is of great interest in computer science. In recent years powerful tools using inductive reasoning have been developed for the case that the specifications admit initial models. Thereby the methods by which inductive reasoning is performed differ, and may be divided into three groups. Firstly, induction is applied explicitly on the structure of terms like in [Bu69] or [BoMo79]. Secondly, completion based techniques for term rewriting systems are used to get so called "proofs by consistency" (cf. [Mu80, HuHu82, Ba88]). Thirdly, a kind of combination of the first and second method is realized as in [KoRu90]. Our approach here adopts the second method heavily relying on the concept of Bachmair in [Ba88].

The specifications to be considered in this paper are parametric, the parameterpart being constituted by an arbitrary (not necissarily equational) theory. On the one hand, this allows to abstract properties by axiomatization that otherwise may be achieved implicitely from bottom-up specifications. On the other hand, the parameter theory can be seperated from the body of the specification and may be treated as built-in. Whereas the syntax of the parameter part is not restricted in any way, the body of the specification has to be equational — precisely it has to consist of positive/negative conditional equations. Hence the specifications we want to analyze differ in various aspects from those already studied by other authors:

Bevers/Lewi in [BeLe91] and Kounalis/Rusinowitch in [KoRu90] have developed methods to prove inductive theorems in positive conditional (bottom-up) specifications. Wirth in [Wi91] allows in addition negative conditions. Parametrized specifications are studied in a general fashion by H. Kirchner (cf. [Ki91]) in the unconditional case where the parameter theory also has to be equational. Ganzinger in [Ga87] considers parametric positive conditional equational specifications which allow also nonequational assertions as "parameter constraints", but he is primarily interested in confluence results and not in inductive theorem proving.

The "particular intended" models of a given specification we are interested in are the canonical term algebras $\mathcal{T}(\mathcal{A})$ induced by the body specification $R$ and the models $\mathcal{A}$ of the parameter theory $T$. By introducing further restrictions on $R$ we achieve that every model $\mathcal{A}$ of the parameter theory $T$ is "contained" (isomorphically embeddable) in the corresponding canonical term algebra $\mathcal{T}(\mathcal{A})$ and that $\mathcal{T}(\mathcal{A})$ is initial in the class of models of R that contain $\mathcal{A}$. Hence the "intended" models are all the "free extensions" (cf. [GoMe87]) $\mathcal{T}(\mathcal{A})$ of the models $\mathcal{A}$ of T induced by R. To be able to define $\mathcal{T}(\mathcal{A})$ we add to R a "description of $\mathcal{A}$" — the diagram of $\mathcal{A}$ — consisting of the "evaluation rules" and the "protection restrictions" induced by $\mathcal{A}$.

Our concept to operationally handle the negative literals that may occur in the conditions of R is influenced by [BaGa91]. We first develope a perfect model semantics and further add conditions until we reach the desired initiality.

The theorems to be proved are clauses that are valid in all the canonical extensions $\mathcal{T}(\mathcal{A})$ of all the models $\mathcal{A}$ of T. Additionally to proving theorems we want to refute hypotheses that are not valid in some $\mathcal{T}(\mathcal{A})$. If the theory T is "simple" we will get a refutationally complete theorem prover.

The paper is organized as follows: The sections can be split into two main parts: the definition of semantics (sections 2 - 5) and theorem proving (sections 6 - 11). Section 2 introduces the "standard situation" we deal with. In section 3 and 4 we define the congruence relation which induces the canonical term model in a slightly more general situation than our standard situation, as these results are of their own interest. In section 5 we transfer the results achieved in the general situation to the standard situation. Sections 6 to 9 are devoted to theorem proving. We introduce a new simplification concept which is influenced by Nieuwenhuis/Orejas (cf. [NiOr91]). In section 10 we illustrate some ideas by examples and in section 11 we present a built-in algorithms for a simple theory.

We assume that the reader is familiar with the basic concepts of term rewriting, equational reasoning (cf. [AvMa90, DeJo90]) and mathematical logic. We define notations only if they differ from standard notations.

# 2 Parametrized Specifications

A *positive/negative conditional specification parametrized by a built-in theory* $(spec_0, spec_1)$ (short: *specification*) consists of $spec_0 = (\Sigma_0, T)$ and $spec_1 = spec_0 + (\Sigma, R)$, where $\Sigma_0$, $\Sigma$ and $\Sigma_1 = \Sigma_0 \cup \Sigma$ are signatures,

$T$ is a set of $\Sigma_0$-axioms of a theory (i.e. a set of closed $\Sigma_0$-formulas) and $R$ is a finite set of positive/negative conditional directed equations over $\Sigma_1$.

A *signature* $\Sigma = (S, F)$ consists of a set $S$ of sort symbols and a set $F$ of function symbols with arities in $S^*$.

A *positive/negative conditional directed equation over* $\Sigma_1$ (short: *conditional equation*) is written in the form $u = v$ *if* $U_1 \wedge \ldots \wedge U_n$ where $u, v$ are $\Sigma_1$-terms and $U_1, \ldots, U_n$ are $\Sigma_1$-literals. As usual we require that $u$ is no variable and that all variables occuring in $v, U_1, \ldots, U_n$ also occur in $u$. We often split the literals of a condition into a positive and a negative part and write them in the form $u_1 = v_1 \wedge \ldots \wedge u_k = v_k \wedge \overline{u}_1 \neq \overline{v}_1 \wedge \ldots \wedge \overline{u}_l \neq \overline{v}_l$ or even $\bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j$.

We write $TERM(F_i, V_i)$ resp. $LIT(F_i, V_i)$ resp $CLAUSE(F_i, V_i)$ for the set of $\Sigma_i$-terms resp. $\Sigma_i$-literals resp. $\Sigma_i$-clauses, where $F_i$ resp. $S_i$ is the set of function symbols resp. sort symbols from $\Sigma_i$ and $V_i$ is the set of variables associated with $S_i$. If we want to express a sort restriction, we add an appropriate index, e.g. $LIT_0(F_1, V_1)$ is the set of $\Sigma_1$-literals $u = v$ resp. $\bar{u} \neq \bar{v}$ over the parameter sorts $S_0$, where $u, v, \bar{u}, \bar{v}$ are $\Sigma_1$-terms of sort $S_0$. We write $TERM(F_i)$ for the set of ground terms over $\Sigma_i$.

The following example, which — in a modified version — is taken from [Gr90], specifies sorting by successively extracting the minimum element. Note that the conditions are $\Sigma_1$-literals over sorts from $S_0$.

**Example 1**

*(a) the signatures*

$$S_0 = \{boole, element\} \qquad S_1 = S_0 \cup \{list\}$$
$$F_0 = \{t, f, >\} \qquad F_1 = F_0 \cup \{nil, ., leq, min, del, ord, sort\}$$

| | | | | |
|---|---|---|---|---|
| $t$ | : | | $\rightarrow$ | $boole$ |
| $f$ | : | | $\rightarrow$ | $boole$ |
| $>$ | : | $element, element$ | $\rightarrow$ | $boole$ |
| $nil$ | : | | $\rightarrow$ | $list$ |
| $.$ | : | $element, list$ | $\rightarrow$ | $list$ |
| $leq$ | : | $element, list$ | $\rightarrow$ | $boole$ |
| $min$ | : | $element, list$ | $\rightarrow$ | $element$ |
| $del$ | : | $element, list$ | $\rightarrow$ | $list$ |
| $ord$ | : | $list$ | $\rightarrow$ | $boole$ |
| $sort$ | : | $list$ | $\rightarrow$ | $list$ |

*(b) the parameter theory* $T = T_{boole} \cup T_{totordset}$

$$T_{boole} = \{t \neq f \ , \ \forall x : boole.\ x = t \vee x = f\}$$
$$T_{totordset} = \{\forall x, y, z : element.\ (x > x \neq t) \wedge (x > y = t \ \wedge \ y > z = t \ \Rightarrow \ x > z = t) \wedge (x > y = t \ \vee \ x = y \ \vee \ y > x = t)\}$$

*(c) the body specification* $R$

| | | | | | |
|---|---|---|---|---|---|
| *(1)* | $leq(x, nil)$ | $=$ | $t$ | | |
| *(2)* | $leq(x, a.A)$ | $=$ | $f$ | *if* | $x > a = t$ |
| *(3)* | $leq(x, a.A)$ | $=$ | $leq(x, A)$ | *if* | $x > a \neq t$ |
| *(4)* | $min(x, nil)$ | $=$ | $x$ | | |
| *(5)* | $min(x, a.A)$ | $=$ | $min(a, A)$ | *if* | $x > a = t$ |
| *(6)* | $min(x, a.A)$ | $=$ | $min(x, A)$ | *if* | $x > a \neq t$ |
| *(7)* | $del(x, nil)$ | $=$ | $nil$ | | |
| *(8)* | $del(x, a.A)$ | $=$ | $del(x, A)$ | *if* | $x = a$ |
| *(9)* | $del(x, a.A)$ | $=$ | $a.del(x, A)$ | *if* | $x \neq a$ |

$$
\begin{array}{lllll}
(10) & ord(nil) & = & t & \\
(11) & ord(a.A) & = & ord(A) & if \quad leq(a,A) = t \\
(12) & ord(a.A) & = & f & if \quad leq(a,A) \neq t
\end{array}
$$

$$
\begin{array}{lll}
(13) & sort(nil) & = nil \\
(14) & sort(a.A) & = min(a,A).sort(del(min(a,A),a.A))
\end{array}
$$

*(d) theorems to be proved*

$$
\begin{array}{ll}
(A) & min(a,A) > a \neq t \\
(B) & leq(min(x,A),A) = t \\
(C) & leq(x,del(a,A)) = leq(x,A) \ \vee \ x > a = t \\
(D) & leq(x,sort(A)) = leq(x,A) \\
(E) & ord(sort(A)) = t
\end{array}
$$

An *actualization of the parameter theory* $T$ is a model $\mathcal{A}$ of T (abbreviated by: $\mathcal{A} \models T$). We want to describe $\mathcal{A}$ by its *diagram*. For that reason we enrich the signatures by adding new constants $\underline{a}$ for each element $a$ of the carrier $A$ of $\mathcal{A}$. Let $C(\mathcal{A})$ be the set of new constants and let $F_i(\mathcal{A}) = F_i \cup C(\mathcal{A})$. The *positive diagram* $DIA^+(\mathcal{A})$ *of* $\mathcal{A}$ is the set of all (directed) equations $f(\underline{a}_1,\ldots,\underline{a}_n) = \underline{a}$ such that $f \in F_0$; $a_i, a \in A$ and $f^{\mathcal{A}}(a_1,\ldots,a_n) = a$. The *negative diagram* $DIA^-(\mathcal{A})$ *of* $\mathcal{A}$ is the set of all negated equations $\underline{a} \neq \underline{b}$ such that $a, b \in A$ are distinct elements of the carrier of $\mathcal{A}$.

If there are negative literals in the conditions one has to use additional informations to get the particular models of interest (cf. [Ka88]). As in [BaGa91] we use a reduction ordering $>$ for that purpose. So let $>$ be always given besides $R$ and $T$. We assume that $R$ is compatible with $>$, i.e for $u = v \; if \bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j \in R$ we have $u > v, u_i, v_i, \overline{u}_j, \overline{v}_j$. To get an ordering that is appropriate for conditional rewrite systems — see the disscussion of decreasing conditional rewrite systems in [DeOkSi88] — we have to require an additional well founded ordering $>_e$ that is an extension of $>$, that contains the proper subterm ordering $>_{st}$ and that is stable w.r.t. substitutions. Now, if $>$ is given, one can always assume $>_e$ to be the transitive closure of $> \cup >_{st}$. For simplicity we will claim in the sequel that the proper subterm ordering $>_{st}$ is even part of $>$ so that we do not need the additional ordering $>_e$.

**Definition 1** *A (reduction) ordering* $>$ *on* $TERM(F_1,V_1)$ *is said to be* T-extendable *iff for every* $\mathcal{A} \models T$ *there is a (reduction) ordering* $>_{(\mathcal{A})}$ *on* $TERM(F_1(\mathcal{A}),V_1)$ *containing* $>$ *such that* $DIA^+(\mathcal{A})$ *is compatible with* $>_{(\mathcal{A})}$.

Note that if $>_{(\mathcal{A})}$ is a $T$-extension of a reduction ordering $>$ which has the proper subterm property, then $>_{(\mathcal{A})}$ has the proper subterm property with respect to the greater set of terms as well. For a proof just exchange the constants from $C(\mathcal{A})$ by new variables.

Finally, if we are given $T, R, >$ with $>$ assumed to be $T$-extendable, we get after actualizing $T$ by $\mathcal{A}$ a triple $(R(\mathcal{A}), N(\mathcal{A}), >_{(\mathcal{A})})$ where

- $R(\mathcal{A}) = R \cup DIA^+(\mathcal{A})$ is the set of equations to be used to construct the term model $\mathcal{T}(\mathcal{A})$.
- $N(\mathcal{A}) = DIA^-(\mathcal{A})$ is a set of "protection"-restrictions.
- $>_{(\mathcal{A})}$ is a reduction ordering which is used as induction guide in the construction of the term model $\mathcal{T}(\mathcal{A})$.

# 3 The canonical term algebra

As mentioned in the introduction, we abstract here from the "standard (parameter passed) situation" — described by a triple $(R(\mathcal{A}), N(\mathcal{A}) >_{(\mathcal{A})})$ — by considering arbitrary triples $(\mathcal{R}, \mathcal{N}, \succ)$ with

- $\succ$ being a reduction ordering on $TERM(F,V)$ with $>_{st} \subseteq \succ$,
- $\mathcal{R}$ being a set of positive/negative conditional directed equations over $\Sigma$ compatible with $\succ$ and

- $\mathcal{N}$ being a set of negated equations over $\Sigma$.

The results obtained for such an arbitrary triple will be carried over to the standard triples in section 5. $\mathcal{R}$ and $\succ$ will be used to construct the term model $\mathcal{T}$ whereas $\mathcal{N}$ serves as a set of restrictions. Note that in contrast to [BaGa91] we do not require $\succ$ to be total on ground terms.

Let $INST(\mathcal{R})$ (resp. $INST(\mathcal{N})$) be the set of ground instances of $\mathcal{R}$ (resp. $\mathcal{N}$), i.e. $INST(\mathcal{R}) = \{\tau(u) = \tau(v)\ if\ \tau(U_1) \wedge \ldots \wedge \tau(U_n)\ \mid\ u = v\ if\ U_1 \wedge \ldots \wedge U_n\ \in \mathcal{R}\ and\ \tau\ is\ a\ ground\ substitution\}$.

We define by noetherian induction on $\succ$ for each $u \in TERM(F)$ a set $G_u$ of unconditional directed ground equations:

Let $u \in TERM(F)$ be a ground term and suppose that $G_v$ is already defined for all ground terms $v$ for which $u \succ v$ holds. Then let

$$G_{\prec u} = \bigcup_{u \succ v} G_v \quad and$$

$$G_u = G_{\prec u} \cup \{u = v \mid there\ is\ an\ instance\ u = v\ if \bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j \in INST(\mathcal{R})\ such\ that$$
$$u_i \overset{*}{\longleftrightarrow}_{\prec u} v_i\ for\ i = 1 \ldots k\ and\ not\ \overline{u}_j \overset{*}{\longleftrightarrow}_{\prec u} \overline{v}_j\ for\ j = 1 \ldots l\}$$

Note that we write $\longrightarrow_{\prec u}$ resp. $\longrightarrow_u$ instead of $\longrightarrow_{G_{\prec u}}$ resp $\longrightarrow_{G_u}$. Finally let

$$G = \bigcup_{u \in TERM(F)} G_u$$

be the ground rewrite system induced by $\mathcal{R}$ and $\succ$.

The rewrite relation $\longrightarrow = \longrightarrow_G$ induces a congruence relation $\sim = \overset{*}{\longleftrightarrow}_G$ on the set $TERM(F)$ of ground terms and henceforth via $\sim$ a canonical term algebra $\mathcal{T}$ with carrier $TERM(F)_{/\sim}$. $\mathcal{T}$ will be called *the canonical term algebra induced by* $(\mathcal{R}, \succ)$.

We want $\mathcal{T}$ to be a model of $\mathcal{R}$ and $\mathcal{N}$ — which is not true in gerenal. To guarantee this fact we introduce the notion of consistency property. First let $G_{\{s,t\}} = \{u = v \in G \mid s \succeq u\ or\ t \succeq u\}$ and as usual $\longrightarrow_{\{s,t\}} = \longrightarrow_{G_{\{s,t\}}}$.

**Consistency properties for $(\mathcal{R}, \mathcal{N}, \succ)$:**
*(c1) For any $s, t \in TERM(F)$ we have: $s \overset{*}{\longleftrightarrow} t$ iff $s \overset{*}{\longleftrightarrow}_{\{s,t\}} t$.*
*(c2) For any $u \neq v \in INST(\mathcal{N})$ we have: not $u \overset{*}{\longleftrightarrow} v$.*

**Lemma 1** *Let $(\mathcal{R}, \mathcal{N}, \succ)$ satisfy the consistency properties (c1) and (c2). Then $\mathcal{T}$ is a model of $\mathcal{R}$ and $\mathcal{N}$.*

**Proof:** $\mathcal{T} \models \mathcal{N}$ is trivial by (c2). Let $u = v\ if \bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j \in INST(\mathcal{R})$ and $\mathcal{T} \models \bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j$. Hence $u_i \overset{*}{\longleftrightarrow} v_i$ and not $\overline{u}_j \overset{*}{\longleftrightarrow} \overline{v}_j$. By (c1) we get $u_i \overset{*}{\longleftrightarrow}_{\{u_i, v_i\}} v_i$. As $u \succ u_i, v_i$ we have $u_i \overset{*}{\longleftrightarrow}_{\prec u} v_i$. If $\overline{u}_j \overset{*}{\longleftrightarrow}_{\prec u} \overline{v}_j$, then $\overline{u}_j \overset{*}{\longleftrightarrow} \overline{v}_j$. Consequently $u = v \in G$ and $\mathcal{T} \models u = v$. □

The operational treatment of negations as defined above leads to so called perfect models. We summarize in short the ideas and results necessary for our context (cf. [BaGa91] for the general case).

**Example 2** $\mathcal{R} = \{c = b\ if\ a \neq b\}$; $\mathcal{N} = \emptyset$; $c \succ b$, $c \succ a$

There are three term models of $\mathcal{R}$ (and $\mathcal{N}$):

- $\mathcal{T}_1$ satisfies $a \neq b = c$.
- $\mathcal{T}_2$ satisfies $a = b \neq c$.
- $\mathcal{T}_3$ satisfies $a = b = c$.

$\mathcal{T}_1$ is the model we prefer, as $\mathcal{T}_1$ is a "minimal" model and as the term equivalences in $\mathcal{T}_1$ are "constructive". We make precise these ideas by introducing the notion of a perfect model.

Arbitrary models of $\mathcal{R}$ $(\cup\mathcal{N})$ can be compared by related congruence relations on $TERM(F)$. Let $\mathcal{A}$ be an algebra (over the signature related to F). $\mathcal{A}$ induces a congruence relation $\sim_{\mathcal{A}}$ on $TERM(F)$ by $s \sim_{\mathcal{A}} t\ iff\ s^{\mathcal{A}} = t^{\mathcal{A}}$, where $s^{\mathcal{A}}$ and $t^{\mathcal{A}}$ are the values of $s$ and $t$ in $\mathcal{A}$. Let $\succ^p = \prec\prec$ be the *preference ordering* on multisets of terms. We compare two congruence relations $\sim_1$ and $\sim_2$ on $TERM(F)$ by

$\sim_1 \succ^i \sim_2$ iff $\{\{s,t\} \mid s \sim_1 t\} \succ^p \succ^p \{\{s,t\} \mid s \sim_2 t\}$.

**Definition 2** *An algebra $\mathcal{A}$ is called a* perfect model of $(\mathcal{R},\mathcal{N},\succ)$ *iff $\mathcal{A}$ is a model of $\mathcal{R}\cup\mathcal{N}$ and for any other model $\mathcal{B}$ of $\mathcal{R}\cup\mathcal{N}$ we have $\sim_{\mathcal{B}}\succeq^{i}\sim_{\mathcal{A}}$.*

**Theorem 1** *Let $(\mathcal{R},\mathcal{N},\succ)$ satisfy the consistency properties (c1) and (c2). Then $\mathcal{T}$ is a perfect model of $\mathcal{R}\cup\mathcal{N}$.*

**Proof:** (c. [BaGa91]) By Lemma 1, $\mathcal{T}\models\mathcal{R}\cup\mathcal{N}$. Now let $\mathcal{B}\models\mathcal{R}\cup\mathcal{N}$. Assume $\sim\neq\sim_{\mathcal{B}}$. Let $\{s,t\}\in\sim\setminus\sim_{\mathcal{B}}$. We construct $\{s',t'\}\in\sim_{\mathcal{B}}\setminus\sim$ with $\{s',t'\}\succ^{p}\{s,t\}$ resp. $\{s,t\}\succ\succ\{s',t'\}$.
Case 1: $\{s,t\}$ is a (w.r.t. $\succ\succ$) minimal element of $\sim\setminus\sim_{\mathcal{B}}$: As $s\overset{*}{\longleftrightarrow}t$ we get by (c1) $s\overset{*}{\longleftrightarrow}_{\{s,t\}}t$ resp. $s\equiv s_0\longleftrightarrow_{\{s,t\}}s_1\longleftrightarrow_{\{s,t\}}\cdots\longleftrightarrow_{\{s,t\}}s_n\equiv t$ for some $n$ and $s_0,\ldots,s_n\in TERM(F)$. As $s\not\sim_{\mathcal{B}}t$ there exists a $m\in\{1,\ldots,n\}$ so that $s_{m-1}\not\sim_{\mathcal{B}}s_m$. Let $s_{m-1}\longleftrightarrow_{\{s,t\}}s_m$ with $u=v\in G_{\{s,t\}}$. By the definition of $G$ there is some $u=v$ *if* $\bigwedge_i u_i=v_i\bigwedge_j\overline{u}_j\neq\overline{v}_j\in INST(R)$ with $u_i\overset{*}{\longleftrightarrow}_{\prec_{\mathcal{A}}}v_i$ and not $\overline{u}_j\overset{*}{\longleftrightarrow}_{\prec_{\mathcal{A}}}\overline{v}_j$. We get $u\not\sim_{\mathcal{B}}v$, for otherwise $s_{m-1}\sim_{\mathcal{B}}s_m$. Further $u_i\sim_{\mathcal{B}}v_i$, for otherwise $\{s,t\}$ would not be a minimal element of $\sim\setminus\sim_{\mathcal{B}}$. We have $\overline{u}_j\sim_{\mathcal{B}}\overline{v}_j$ for some $j\in\{1,\ldots,l\}$, for otherwise $u\sim_{\mathcal{B}}v$ as $\mathcal{B}\models\ u=v$ *if* $\bigwedge_i u_i=v_i\bigwedge_j\overline{u}_j\neq\overline{v}_j$.
Finally $\overline{u}_j\not\sim\overline{v}_j$, for otherwise $\overline{u}_j\overset{*}{\longleftrightarrow}_{\{\overline{u}_j,\overline{v}_j\}}\overline{v}_j$ by (c1) and hence $\overline{u}_j\overset{*}{\longleftrightarrow}_{\prec_{\mathcal{A}}}\overline{v}_j$. We get $\{\overline{u}_j,\overline{v}_j\}\in\sim_{\mathcal{B}}\setminus\sim$ and $\{s,t\}\succ\succ\{\overline{u}_j,\overline{v}_j\}$. Hence let $s'\equiv\overline{u}_j$ and $t'\equiv\overline{v}_j$.
Case 2: $\{s,t\}$ is not a minimal element of $\sim\setminus\sim_{\mathcal{B}}$: As $\succ\succ$ is noetherian there exists a w.r.t. $\succ\succ$ minimal element $\{s'',t''\}$ in $\sim\setminus\sim_{\mathcal{B}}$ such that $\{s,t\}\succ\succ\{s'',t''\}$. By case 1 we get $\{s',t'\}\in\sim_{\mathcal{B}}\setminus\sim$ with $\{s'',t''\}\succ\succ\{s',t'\}$ and consequently $\{s,t\}\succ\succ\{s',t'\}$. □

Example 2 above shows that a perfect model need not be initial. Now we want $\mathcal{T}$ not only to be a perfect but also an initial model of $\mathcal{R}\cup\mathcal{N}$. We know that $\mathcal{T}$ is initial in the class of algebras $\mathcal{B}$ for which $\sim$ is a subset of $\sim_{\mathcal{B}}$. By requiring that certain inequivalences are equivalently transformable into "restrictions" in $\mathcal{N}$ we will be able to prove that $\sim\subseteq\sim_{\mathcal{B}}$ for any $\mathcal{B}\models\mathcal{R}\cup\mathcal{N}$.

Let $\mathcal{N}_{\sim}$ be the set of negated equations $u\neq v$ $(u,v\in TERM(F,V))$ such that for any ground substitution $\tau$ with $\tau(u)\not\sim\tau(v)$ there is an instance $a\neq b\in INST(\mathcal{N})$ (of a "restriction") with $\tau(u)\sim a$, $\tau(v)\sim b$, $\tau(u)\succeq a$ and $\tau(v)\succeq b$. We might say that inequivalences that result from instantiating a member of $\mathcal{N}_{\sim}$ are operationally covered by the "restrictions" $\mathcal{N}$.

Let $\mathcal{N}_{\mathcal{R}}$ be the set of negativ literals that occur in the conditions of the conditional equations in $\mathcal{R}$.

**Negation covering property for $(\mathcal{R},\mathcal{N},\succ)$:**
*(nc)* $\mathcal{N}_{\mathcal{R}}\subseteq\mathcal{N}_{\sim}$

**Lemma 2** *Let $(\mathcal{R},\mathcal{N},\succ)$ satisfy the consistency properties (c1) and (c2) as well as the negation covering property (nc). Then $\sim\subseteq\sim_{\mathcal{B}}$ for any model $\mathcal{B}$ of $\mathcal{R}\cup\mathcal{N}$.*

**Proof:** Let $\mathcal{B}\models\mathcal{R}\cup\mathcal{N}$. By noetherian induction w.r.t. $\succ\succ$ we prove $P(\{s,t\})$ for any $s,t\in TERM(F)$ with $P(\{s,t\})$ iff $s\sim t$ implies $s\sim_{\mathcal{B}}t$. Assume $P(\{s',t'\})$ for any $s',t'\in TERM(F)$ with $\{s,t\}\succ\succ\{s',t'\}$. Let $s\sim t$. Suppose $s\not\sim_{\mathcal{B}}t$. Then — check the proof of theorem 1 — we get $\bar{u}_j\neq\bar{v}_j\in INST(\mathcal{N}_{\mathcal{R}})$ with $\{\bar{u}_j,\bar{v}_j\}\in\sim_{\mathcal{B}}\setminus\sim$ and $\{s,t\}\succ\succ\{\bar{u}_j,\bar{v}_j\}$. By (nc) there is $a\neq b\in INST(\mathcal{N})$ with $\bar{u}_j\sim a$ and $\bar{v}_j\sim b$ as well as $\bar{u}_j\succeq a$ and $\bar{v}_j\succeq b$. By the induction hypothesis we get $\bar{u}_j\sim_{\mathcal{B}}a$ and $\bar{v}_j\sim_{\mathcal{B}}b$. Hence $a\sim_{\mathcal{B}}b$ which contradicts $\mathcal{B}\models\mathcal{N}$. □

# 4 Sufficient criteria for the consistency properties

We assume that $(\mathcal{R},\mathcal{N},\succ)$ is given as in section 3. We use the same notations as above.

**Lemma 3** *If $\longrightarrow$ is confluent, then $(\mathcal{R},\mathcal{N},\succ)$ satisfies the consistency property (c1).*

**Lemma 4** *If $\longrightarrow$ is confluent and if $a$ and $b$ are neither identical nor $\longrightarrow$-reducible for any instance $a\neq b\in INST(\mathcal{N})$, then $(\mathcal{R},\mathcal{N},\succ)$ satisfies the consistency property (c2).*

We now turn to the question of how to guarantee the confluence of $\longrightarrow$, which plays the central role in the lemmata above. The rewrite system $G$ is not given directly, but has to be constructed by an inductive process. So how can we make statements about $G$ by only inspecting $\mathcal{R}$? To give an answer to this question we introduce the new rewrite relation $\longrightarrow_{\mathcal{R}}$:

**Definition 3** *For $s,t \in TERM(F,V)$ let $s \longrightarrow_{\mathcal{R}} t$ iff there exists an occurence $p \in O(s)$, a substitution $\sigma$ and an equation $u = v$ if $\bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j \in \mathcal{R}$ such that $s/p \equiv \sigma(u)$, $s[p \leftarrow \sigma(v)] \equiv t$ and $\sigma(u_i) \downarrow_{\mathcal{R}} \sigma(v_i)$ and not $\sigma(\overline{u}_j) \downarrow_{\mathcal{R}} \sigma(\overline{v}_j)$.*

**Lemma 5** *Let $t \in TERM(F)$ and let $\longrightarrow_s$ be confluent for all $s \prec t$. Then for all $s \prec t$ and all $s'$ we have: $s \longrightarrow_s s'$ iff $s \longrightarrow_{\mathcal{R}} s'$.*

**Proof:** by inoetherian induction

**Theorem 2** *If $\longrightarrow_{\mathcal{R}}$ is ground confluent, then $\longrightarrow_G$ is confluent.*

**Proof:** We prove by noetherian induction on $\succ$ that $\longrightarrow_t$ is confluent for all $t \in TERM(F)$. Then the claim follows immediately. Let $t \in TERM(F)$ and assume that $\longrightarrow_s$ is confluent for all $s \prec t$. As $\longrightarrow_G$ is noetherian (note that $\mathcal{R}$ and hence $G$ is compatible with $\succ$) it suffices to prove local confluence and — as usual — to consider critical divergencies $v \;{}_t\!\longleftarrow u \longrightarrow_t u[p \leftarrow v']$ with $u = v, u' = v' \in G_t$ and $u/p \equiv u'$.
We first want to prove $v \;{}_{\mathcal{R}}\!\longleftarrow u \longrightarrow_{\mathcal{R}} u[p \leftarrow v']$. If $t \succ u$ and $t \succ u'$, then we can use Lemma 5 and the induction hypothesis. Now let w.l.o.g. $t \not\succ u$. Then $t \equiv u$ as $u = v \in G_t$. We know that there exists some $u = v$ *if* $\bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j \in INST(\mathcal{R})$ with $u_i \overset{*}{\longleftrightarrow}_{\prec u} v_i$ and not $\overline{u}_j \overset{*}{\longleftrightarrow}_{\prec u} \overline{v}_j$. By assumption $\longrightarrow_{\prec u}$ is confluent. Hence $u_i \downarrow_{\prec u} v_i$. As $t \equiv u \succ u_i, v_i$ we get by Lemma 5 $u_i \downarrow_{\mathcal{R}} v_i$. If we had $\overline{u}_j \downarrow_{\mathcal{R}} \overline{v}_j$, then we would get with Lemma 5 $\overline{u}_j \overset{*}{\longleftrightarrow}_{\prec u} \overline{v}_j$. So $u \longrightarrow_{\mathcal{R}} v$. Analogously we get $u \longrightarrow_{\mathcal{R}} u[p \leftarrow v']$.
As $\longrightarrow_{\mathcal{R}}$ is ground confluent by assumption there exists a $w \in TERM(F)$ with $v \overset{*}{\longrightarrow}_{\mathcal{R}} w \;{}_{\mathcal{R}}\overset{*}{\longleftarrow} u[p \leftarrow v']$. Again by Lemma 5 we can conclude $v \overset{*}{\longrightarrow}_t w \;{}_t\overset{*}{\longleftarrow} u[p \leftarrow v']$. □

**Corollary 1** *If $\longrightarrow_{\mathcal{R}}$ is ground confluent, then for all $s,t \in TERM(F)$ we have: $s \longrightarrow_G t$ iff $s \longrightarrow_{\mathcal{R}} t$.*

Now confluence of positive/negative conditional rewrite systems can be tested in the same manner as in the merely positive conditional case.

A *critical equation* between two (variable disjoint) conditional equations $u = v$ *if* $U$ and $u' = v'$ *if* $U'$ is a conditional equation $\mu(v) = \mu(u)[p \leftarrow \mu(v')]$ *if* $\mu(U) \wedge \mu(U')$ where $u/p$ is no variable, $u/p$ and $u'$ are unifiable and $\mu$ is the most general unifier of $u/p$ and $u'$.

A substitution $\sigma$ *satifies a condition* $\bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j$ *w.r.t.* $\mathcal{R}$ iff $\sigma(u_i) \downarrow_{\mathcal{R}} \sigma(v_i)$ $(i = 1 \ldots k)$ and not $\sigma(\overline{u}_j) \downarrow_{\mathcal{R}} \sigma(\overline{v}_j)$ $(j = 1 \ldots l)$.

A conditional equation $u = v$ *if* $\bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j$ is called *joinable w.r.t.* $\mathcal{R}$ *(and the signature* $\Sigma$*)* iff for any $\Sigma$-substitution $\sigma$ which satisfies $\bigwedge_i u_i = v_i \bigwedge_j \overline{u}_j \neq \overline{v}_j$ we have $\sigma(u) \downarrow_{\mathcal{R}} \sigma(v)$.

**Theorem 3** *If all critical equations that can be built from the conditional equations of $\mathcal{R}$ are joinable w.r.t. $\mathcal{R}$ (and $\Sigma$), then $\mathcal{R}$ is confluent.*

**Proof:** analogous to the positive conditional case

# 5 Canonical term models for theory actualizations

We return to the situation described at the end of section 2 by $R(\mathcal{A})$, $N(\mathcal{A})$ and $>_{(\mathcal{A})}$. For each actualization $\mathcal{A}$ of $T$ we get the "standard situation" $(\mathcal{R}, \mathcal{N}, \succ)$ of section 3 and 4 where $\mathcal{R} = R(\mathcal{A})$, $\mathcal{N} = N(\mathcal{A})$ and $\succ = >_{(\mathcal{A})}$. Hence we get — by the process developed for an arbitrary $(\mathcal{R}, \mathcal{N}, \succ)$ — for each $\mathcal{A} \models T$ a canonical term algebra $\mathcal{T}(\mathcal{A})$.

**Definition 4** *A $\Sigma_1(\mathcal{A})$-algebra $\mathcal{B}$* contains *the $\Sigma_0$-algebra $\mathcal{A}$ iff the $\Sigma_0$-algebra induced by $\{\underline{a}^{\mathcal{B}} \mid a \in A\}$ is isomorphic to $\mathcal{A}$ via $i : a \mapsto \underline{a}^{\mathcal{B}}$.*

In the next lemma we put together the results of lemma 1 and lemma 2 of section 3 in the context of the parameter passed situation.

**Lemma 6** *For any $\mathcal{A} \models T$, if $(R(\mathcal{A}), N(\mathcal{A}), >_{(\mathcal{A})})$ satisfies the conditions (c1), (c2) and (nc), then $T(\mathcal{A})$ is initial in the class of $\Sigma_1(\mathcal{A})$-algebras that contain $\mathcal{A}$ and satisfy $R$.*

Next we will formulate sufficient criteria, such that $(R(\mathcal{A}), N(\mathcal{A}), >_{(\mathcal{A})})$ satisfies (c1), (c2) and (nc) for all $\mathcal{A} \models T$. Note that the criteria should be independent of the special models $\mathcal{A}$.

**Definition 5** *$R$ is said to be* $T$-(ground) confluent *iff $R(\mathcal{A})$ is (ground) confluent for all $\mathcal{A} \models T$.*

We want $R$ to be $T$-ground confluent. For that reason we restrict $R$ in a rather strong manner, such that there will be no critical overlaps between equations from $R$ and equations from $DIA^+(\mathcal{A})$ for any $\mathcal{A} \models T$.

**Definition 6** *$R$ is called* parameter simple *iff the left hand sides of the (directed) equations of $R$ do not contain any symbol from $F_0$.*

Note that $R$ of example 1 is parameter simple. We want to use theorem 3 to conclude the confluence of $R(\mathcal{A})$ for any $\mathcal{A} \models T$. But in general joinability of the critical equations (that only result from $R$ if $R$ is parameter simple) can not be treated with $R$ alone. One has to take into account the additional equations from $DIA^+(R)$ too, as the next example demonstrates.

**Example 3** $F_0 = \{b, c, d, e\}$; $F_1 = F_0 \cup \{a\}$; $a > b, c, d, e$; $T = \emptyset$

$R$: $a = b$
$a = c$ *if* $d = e$

$\mathcal{A} \models (d = e) \wedge (b \neq c)$; $\mathcal{B} \models (d \neq e)$

$R$ is parameter simple. Whereas $R(\mathcal{B})$ is (ground) confluent (because the condition $d = e$ of the critical equation $b = c$ *if* $d = e$ can not be satisfied by a $F_1(\mathcal{B})$-substitution) $R(\mathcal{A})$ is not (by a similar argument). The example shows that satisfiability of a condition in general depends on the model of $T$. The next example indicates that the reduction relation also depends on the model of $T$.

**Example 4** $F_0 = \{b, c\}$; $F_1 = F_0 \cup \{a\}$; $a > b, c$; $T = \emptyset$

$R$: $a = b$ *if* $b \neq c$

$\mathcal{A} \models (b = c)$; $\mathcal{B} \models (b \neq c)$

The term $a$ can be rewritten by $R(\mathcal{B})$ into $b$ wheras this is impossible by $R(\mathcal{A})$.

**Definition 7** *A conditional equation is said to be* $T$-(ground) joinable *iff it is (ground) joinable w.r.t. $R(\mathcal{A})$ and $\Sigma_1(\mathcal{A})$ for any $\mathcal{A} \models T$. A condition is said to be* $T$-unreachable *iff for any $\mathcal{A} \models T$ there exists no $\Sigma_1(\mathcal{A})$-substitution that satisfies the condition w.r.t. $R(\mathcal{A})$.*

Obviously, if the condition of a conditional equation is $T$-unreachable, then the conditional equation is $T$-joinable. If a condition contains two complementary literals $u = v$ and $u \neq v$ (as it is the case in the critical equations of example 1), then the condition is $T$-unreachable for any theory $T$.

**Corollary 2** *Let $R$ be parameter simple. If all critical equations from $R$ are $T$-(ground) joinable, then $R$ is $T$-(ground) confluent.*

**Corollary 3** *Let $R$ be parameter simple. If all critical equations from $R$ are $T$-ground joinable, then $(R(\mathcal{A}), N(\mathcal{A}), >_{(\mathcal{A})})$ satisfies the consistency properties (c1) and (c2) for any $\mathcal{A} \models T$.*

*Proof:* Note that for any $\underline{a} \neq \underline{b} \in N(\mathcal{A})$ $(\mathcal{A} \models T)$ $\underline{a}$ and $\underline{b}$ are distinct constants and irreducible in $G(\mathcal{A})$. The conclusion then follows immediately by previous lemmata.□

Next we give a sufficient criterion for $(R(\mathcal{A}), N(\mathcal{A}), >_{(\mathcal{A})})$ to satisfy (nc) for all $\mathcal{A} \models T$.

**Definition 8** *$R$ is called* $T$-sufficient complete w.r.t. parameter sorts *iff for any $\mathcal{A} \models T$ and any ground term $t \in TERM_0(F_1(\mathcal{A}))$ of a parameter sort (this is indicated by the index 0) there exists a constant $\underline{a} \in C(\mathcal{A})$ such that $t \sim_{(\mathcal{A})} \underline{a}$.*

**Lemma 7** *Let $R$ be a set of conditional equations such that the literals in the conditions are $\Sigma_1$-literals over the parameter sorts. Let $R$ be $T$-ground confluent and $T$-sufficient complete w.r.t. parameter sorts. Then $(R(\mathcal{A}), N(\mathcal{A}), >_{(\mathcal{A})})$ satisfies the negation covering property (nc) for any $\mathcal{A} \models T$.*

*Proof:* Let $u \neq v \in N_R$ and $\tau$ be a ground substitution with $\tau(u) \not\sim_{(\mathcal{A})} \tau(v)$. By the assumptions there exist $\underline{a}$ and $\underline{b}$ in $C(\mathcal{A})$ such that $\tau(u) \sim_{(\mathcal{A})} \underline{a}$ and $\tau(v) \sim_{(\mathcal{A})} \underline{b}$. By lemma 1 and corollary 1 we get $\tau(u) \xrightarrow{*}_{R(\mathcal{A})} \underline{a}$ and $\tau(v) \xrightarrow{*}_{R(\mathcal{A})} \underline{b}$. Hence $\tau(u) \geq_{(\mathcal{A})} \underline{a}$ and $\tau(v) \geq_{(\mathcal{A})} \underline{b}$ as $>$ is $T$-extendable by assumption. If $\underline{a} \neq \underline{b}$ were not an element of $N(\mathcal{A})$, then $\underline{a}$ and $\underline{b}$ would be the same constant and hence $\tau(u) \sim_{(\mathcal{A})} \tau(v)$. □

We do not describe a test for T-sufficient completeness here. A simple test would be to check whether some kind of syntactical case splitting — like in example 1 — is possible.

Though the sort restriction concerning the conditions seems rather restrictive, in practice it is not if a "boolean theory" is available. Most conditions are expressed with the aid of newdefined predicates (i.e. functions of some sort "boole"). We could weaken the sort restriction by allowing positive literals over arbitrary sorts in the conditions, but this would cause other difficulties which we will discuss in the next section.

We finish this section with some remarks about $T$-extendable reduction orderings. If $>$ is a recursive (lexicographic) path ordering (cf. [De87]), then $>$ is $T$-extendable for an arbitrary theory $T$. Just extend the precedence $>_{F_1}$ on $F_1$ to a precedence $>_{F_1(\mathcal{A})}$ on $F_1(\mathcal{A})$ by $>_{F_1(\mathcal{A})} = >_{F_1} \cup \{(f, \underline{a}) \mid f \in F_1 \text{ and } \underline{a} \in C(\mathcal{A})\}$. But not every reduction ordering is $T$-extendable for every theory $T$ as is shown in the example to follow.

**Example 5** $F_0 = \{b, c\}$; $\quad F_1 = F_0 \cup \{f, g\}$

$R$: $g(x, x) = f(b, c)$
$\quad f(x, x) = g(x, x)$

$> \;=\; \xrightarrow{+}_R$

$>$ is a reduction ordering. Let $T = \emptyset$ and $\mathcal{A}$ be an $F_1$-algebra with $\mathcal{A} \models (b = c)$. Hence there is an $a \in A$ such that $b = \underline{a}, c = \underline{a} \in DIA^+(\mathcal{A})$. If $>_{(\mathcal{A})}$ were a $T$-extension of $>$ then:

$$g(\underline{a}, \underline{a}) >_{(\mathcal{A})} f(b, c) >_{(\mathcal{A})} f(\underline{a}, c) >_{(\mathcal{A})} f(\underline{a}, \underline{a}) >_{(\mathcal{A})} g(\underline{a}, \underline{a})$$

The fact that a parameter variable occurs twice in the left hand side of an equation of $R$ is crucial as is shown in the next lemma (stated without proof).

**Definition 9** *$R$ is said to be* parameter leftlinear *iff no variable of a parameter sort occurs more than once in the left hand sides of the equations of $R$.*

**Lemma 8** *If $>$ is a reduction ordering that is induced by a parameter simple and parameter leftlinear unconditional rewrite system $R$ (i.e. $>= \xrightarrow{+}_R$), then $>$ is $T$-extendable for any theory $T$.*

# 6 Proving $T$-inductive theorems

In the sections to follow we study inductive theorem proving w.r.t. specifications that are parametrized by a built-in theory. Let $R$, $>$ and $T$ be given as in the last section, especially, let $R$ be a set of conditional equations with the conditions being over parameter sorts. We require $R$ to be $T$-ground confluent and $T$-sufficient complete w.r.t. parameter sorts in order to get initial models $T(\mathcal{A})$ of $R$ containing $\mathcal{A}$ for any model $\mathcal{A}$ of $T$. Theorems to be proved are $\Sigma_1$-clauses.

**Definition 10**
*(a) A $\Sigma_1$-clause $\Gamma$ is a* $T$-inductive theorem w.r.t. $R$ *iff $T(\mathcal{A})$ is a model of $\Gamma$ for any $\mathcal{A} \models T$. Let $ITH(T;R)$ denote the set of $T$-inductive theorems w.r.t. $R$. An element resp. subset of $ITH(T;R)$ is also said to be* inductively valid w.r.t. $R$ and $T$.
*(b) A $\Sigma_0$-clause $\Gamma$ is said to be* parameter valid w.r.t. $T$ *iff $\Gamma$ is a logical consequence of $T$ (i.e. any model of $T$ is also a model of $\Gamma$).*

First we want to achieve a theorem prover that is refutational complete modulo $T$, i.e.: a set of $\Sigma_1$-clauses that is not inductively valid w.r.t. $R$ and $T$ should be transformable in a finite number of steps into a set of $\Sigma_1$-clauses containing a $\Sigma_0$-clause that is not parameter valid w.r.t. $T$. The theorem prover becomes (fully) refutational complete, if there exists a (built-in parameter) algorithm to decide whether a $\Sigma_0$-clause is parameter valid w.r.t. $T$ or not. In order to be able to transform arbitrary clauses into $\Sigma_0$-clauses we only consider $\Sigma_1$-clauses over parameter sorts. When speaking in the sequel of clauses we always mean $\Sigma_1$-clauses over $S_0$.

Note that by our assumptions we solve the problem of testing inductive reducability — which is shown to be undecidable in general in the conditional case in [KaCh86] — by trivializing it. Nevertheless we may not be able to decide whether a clause is a $T$-inductive theorem if the parameter validity w.r.t. $T$ is undecidable.

The next lemma states the relationship between the semantically defined property of inductive validity w.r.t. $R$ and $T$ and the operationally defined congruence relation. We use $\sigma(\bigvee_i u_i = v_i \bigvee_j \overline{u}_j \neq \overline{v}_j) \xrightarrow{*}_{R(\mathcal{A})}$ *false* as an abbreviation for the fact that for all $i \in \{1,\ldots,k\}$ we have $\sigma(u_i) \downarrow_{R(\mathcal{A})} \neq \sigma(v_i) \downarrow_{R(\mathcal{A})}$ and for all $j \in \{1,\ldots,l\}$ we have $\sigma(\overline{u}_j) \downarrow_{R(\mathcal{A})} = \sigma(\overline{v}_j) \downarrow_{R(\mathcal{A})}$ ($\bigvee_i u_i = v_i \bigvee_j \overline{u}_j \neq \overline{v}_j$ is a clause and $\sigma$ a $\Sigma_1(\mathcal{A})$-ground substitution). Note that $\longrightarrow_{R(\mathcal{A})}$ is ground convergent by our assumptions.

**Lemma 9** *Let $R$ be $T$-ground confluent and $T$-sufficient complete w.r.t. parameter sorts. Let $\Gamma$ be a $\Sigma_1$-clause over $S_0$. Then $\Gamma$ is not inductively valid w.r.t. $R$ and $T$ iff for some $\mathcal{A} \models T$ and some $\Sigma_1(\mathcal{A})$-ground substitution $\sigma$ we have $\sigma(\Gamma) \xrightarrow{*}_{R(\mathcal{A})}$ false.*

**Lemma 10** *Let $R$ be $T$-ground confluent and $T$-sufficient complete w.r.t. parameter sorts. Then, if a $\Sigma_0$-clause is not parameter valid w.r.t. $T$, then it is not inductively valid w.r.t. $R$ and $T$.*

The assumptions being clarified we return to the theorem prover. It is given by an inference system together with a strategy which produce the negative statement "no theorem" if some clause is present that is not inductively valid w.r.t. $R$ and $T$ and which allow to make the positive statement "the clauses are theorems" by absence of such clauses, but which possibly does not terminate in the latter case. The rules of the inference system can be described by "reduction-rules" and "coverage-expansion-rules". For an efficient prover it is important to reduce as much as possible. But reduction with conditional equations causes the problem of how to treat the conditions. One either may verify them — possibly after case splitting — before reducing or one may defer the verification and carry them along as additional context. [KoRu90], [BeLe91] and [Ga87] choose the first possibility whereas we adopt like [Wi91] the second one, as we think it to be the more general one.

To make understandable the notions below we give a short motivation. By reducing a clause $\Gamma$ contextually (in the sense of [ZaRe85]) with a conditional equation $u = v$ *if* $U_1 \wedge \ldots \wedge U_n$ one gets in the rewrite case (see below) a rewritten clause of the form $\sigma(U_1 \wedge \ldots \wedge U_n) \Rightarrow \Gamma'$ resp. $\neg\sigma(U_1) \vee \ldots \vee \neg\sigma(U_n) \vee \Gamma'$ and "rest clauses" of the form $\sigma(\neg U_i) \Rightarrow \Gamma$ resp. $\sigma(U_i) \vee \Gamma$. We want all resulting clauses to be "smaller" than $\Gamma$. But if we introduce complexities as usual by the multiset of the complexities of the literals constituting the clauses, we do not get "smaller" results in the rest clause case. The idea to overcome this situation — inspired by [NiOr91] — can be described as follows:
$\sigma(U_i)$ is "smaller" than some literal in $\Gamma$ resp. it is a member of the so called $<$-complement of $\Gamma$. If $\sigma(U_i)$ does not occur in $\Gamma$ or, equivalently, if $\neg\sigma(U_i) \vee \Gamma$ is not "apparently tautological", then the $<$-complement of $\sigma(U_i) \vee \Gamma$ is "smaller" than the $<$-complement of $\Gamma$. But by switching over to the $<$-complement instead of the clause itself we get problems with the rewritten clause. If we make a literal $L$ occuring in $\Gamma$ "smaller", then the $<$-complement may "increase". To overcome this problem the literal $L$ will not only be substituted by a new $L'$ but also conserverd (if necessary) in an additional clause that we will call *history clause*.
Syntactically clauses will be split into two parts written as $\Gamma//\Phi$, the clause of primary interest $\Gamma$ and the assistant history clause $\Phi$. We identify a clause $\Gamma$ with $\Gamma//$ if no history clause is present.

We make precise some notions used throughout the rest of the paper.
We identify a literal $u = v$ resp. $u \neq v$ with the multiset $\{u, v\}$ and a clause with the multiset of the literals (regarded as multisets) constituting the clause. We identify clauses that are — up to a variable permutation — equivalent as multisets. We use the symbol $\approx$ for the equivalence of multisets.

Let $\top$ be an extra symbol denoting a literal that is valid in every algebra. Let $\gg$ (the multiset extension of $>$) be extended such that $L \gg \top$ for any literal $L$ (we will use the same symbol for the extension).

A clause $\Gamma = C_1 \vee \ldots \vee C_n$ *majorizes* a clause $\Delta = D_1 \vee \ldots \vee D_k$ iff for any index $i \in \{1, \ldots, k\}$ there is a $j \in \{1, \ldots, n\}$ such that $C_j \gg D_i$. The *$<$-complement of a clause* $\Gamma$ — written $cpl_<(\Gamma)$ — is the multiset of literals (each one occuring once) that are majorized by $\Gamma$ and do not occur in $\Gamma$. A clause $\Gamma//\Phi$ is called *majorizing* iff $\Gamma$ majorizes $\Phi$. A literal $C$ in a clause $\Gamma$ is said to be *maximal* iff $C$ is not majorized by $\Gamma \backslash \{C\}$.

By $< u = v > \vee U_1 \vee \ldots \vee U_n$ resp. $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l$ we indicate a clause with one literal resp. several literals singleld out. We call $< u = v > \vee U_1 \vee \ldots \vee U_n$ *rewrite reductive w.r.t.* $\sigma$ iff $\sigma(u) > \sigma(v)$ and $\{\sigma(u)\} \gg \sigma(U_i)$ for $i = 1 \ldots n$. We similarily call $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l$ *subsumption reductive w.r.t.* $\sigma$ iff $\sigma(U_1 \vee \ldots \vee U_k)$ majorizes $\sigma(V_1 \vee \ldots \vee V_l)$. An equation $u = v$ *if* $U_1 \wedge \ldots \wedge U_n$ will also be written in the form $< u = v > \vee \neg U_1 \vee \ldots \vee \neg U_n$.

A clause $\Gamma = C_1 \vee \ldots \vee C_n$ is called *tautological* iff there exist two indices $i, j \in \{1, \ldots, n\}$ such that $C_i \equiv \neg C_j$ (where $\neg u = v \equiv u \neq v$ and $\neg u \neq v \equiv u = v$).

A *critical clause between* $\Gamma//\Phi \equiv C_1 \vee \ldots \vee C_n//\Phi$ *and* $< u = v > \vee U \in R$ *at the literal* $C_i$ is a clause $\Gamma'//\Phi'$ where $\Gamma' \equiv \mu(C_1) \vee \ldots \vee \mu(C_i)[p \leftarrow \mu(v)] \vee \ldots \vee \mu(C_n) \vee \mu(U)$ and $\Phi' \equiv (\mu(\Phi) \vee \mu(C_i)) \cap cpl_<(\Gamma')$ and $C_i/p$ and $u$ are unifiable with mgu $\mu$ and where $C_i/p$ is no variable.

We write $EQ(T)$ for the set of unconditional equations of $T$ and $EQNEG(\bigvee_i u_i = v_i \bigvee_j \overline{u}_j \neq \overline{v}_j)$ for the set $\{\overline{u}_j = \overline{v}_j \mid j = 1 \ldots l\}$ of equations induced by the negative literals in the clause.

# 7 The inference system

The inference system acts upon two sets of clauses, a set $L$ of $T$-inductive theorems — the lemmata — and a set $H$ of splitted clauses — the hypotheses.

(DedHyp — deduction of a hypothesis)

$$\frac{L, H}{L, H \cup \{\Gamma//\Phi\}}$$

if $\Gamma//\Phi$ is a critical clause between an element from $H$ and $R$ at a maximal literal.

(IntLem — introduction of a lemma)

$$\frac{L \quad , H}{L \cup \{\Gamma\}, H}$$

if $\Gamma \in ITH(T; R)$.

(DelTau — deletion of a tautology)

$$\frac{L, H \cup \{\Gamma//\Phi\}}{L, H}$$

if $\Gamma \equiv \top$, or $\Gamma$ contains a literal of the form $u = u$, or $\Gamma$ contains two complementary literals $u = v$ and $u \neq v$.

(DelDed — deletion by built-in deduction)

$$\frac{L, H \cup \{\Gamma \vee \Delta//\Phi\}}{L, H}$$

if $\Gamma \in CLAUSE(F_0, V_0)$ and $\Gamma$ is a logical consequence of $T$.

(DelSub — deletion by $T$-subsumption)

$$\frac{L, H \cup \{\Gamma \vee \Delta // \Phi\}}{L, H}$$

if for some substitution $\sigma$ and some $\Gamma' \in T$ we have $\Gamma = \sigma(\Gamma')$.

(DelEqu — deletion by equivalence transformations)

$$\frac{L, H \cup \{u = v \vee \Gamma // \Phi\}}{L, H}$$

if $u \overset{*}{\longleftrightarrow}_E v$ where $E = EQ(T) \cup EQNEG(\Gamma)$ and the variables in $EQNEG(\Gamma)$ are considered as constants.

(SimpEqu — simplification by equivalence transformations)

$$\frac{L, H \cup \{C \vee \Gamma // \Phi\}}{L, H \cup \{C' \cup \Gamma // \Phi'\}}$$

if $C/p \equiv u$, $C' \equiv C[p \leftarrow v]$, $u \overset{*}{\longleftrightarrow}_E v$ where $E = EQ(T) \cup EQNEG(\Gamma)$, $u > v$ and $\Phi' \equiv (\Phi \cup \{C\}) \cap cpl_<(C' \vee \Gamma)$.

(SimpElim — simplification by literal elimination)

$$\frac{L, H \cup \{C \vee \Gamma // \Phi\}}{L, H \cup \{\Gamma // \Phi'\}}$$

if $C$ occurs in $\Gamma$, or $C \equiv u \neq u$, or $C \in LIT(F_0, V_0)$ and $T \models \neg C$, or $C \equiv \sigma(C')$ and $\neg C' \in T$, and $\Phi' \equiv (\Phi \cup \{C\}) \cap cpl_<(\Gamma)$.

(SimpRew — simplification by clausal rewriting)

$$\frac{L, H \cup \{C \vee \Gamma // \Phi\}}{L, H \cup \{C' \vee \Gamma \vee \sigma(U_1) \vee \ldots \vee \sigma(U_n) // \Phi' \quad , \quad C \vee \Gamma \vee \neg\sigma(U_1) // \Phi \quad , \ldots , \quad C \vee \Gamma \vee \neg\sigma(U_n) // \Phi\}}$$

if there is a substitution $\sigma$, a position p and a clause $< u = v > \vee U_1 \vee \ldots \vee U_n // \Psi$ such that one of the following items holds:

- $< u = v > \vee U_1 \vee \ldots \vee U_n // \Psi \in R \cup L$, $C/p \equiv \sigma(u)$, $C' \equiv C[p \leftarrow \sigma(v)]$, $< u = v > \vee U_1 \vee \ldots \vee U_n$ is rewrite reductive w.r.t. $\sigma$, $\Gamma \vee \Phi \vee \sigma(U_1) \vee \ldots \vee \sigma(U_n)$ is not tautological, $\Phi' \equiv (\Phi \cup \{C\}) \cap cpl_<(C' \vee \Gamma)$.
- $< u = v > \vee U_1 \vee \ldots \vee U_n // \Psi \in H$, $C/p \equiv \sigma(u)$, $C/p \rhd u$ ($\rhd$ is the subsumption ordering), $C' \equiv C[p \leftarrow \sigma(v)]$, $< u = v > \vee U_1 \vee \ldots \vee U_n$ is rewrite reductive w.r.t. $\sigma$, $\Gamma \vee \Phi \vee \sigma(U_1) \vee \ldots \vee \sigma(U_n)$ is not tautological, $\Phi' \equiv (\Phi \cup \{C\}) \cap cpl_<(C' \vee \Gamma)$.

(SimpSub — simplification by clausal subsumption)

$$\frac{L, H \cup \{\Gamma \vee \Delta // \Phi\}}{L, H \cup \{\top \quad , \quad \Gamma \vee \Delta \vee \neg\sigma(V_1) // \Phi \quad , \ldots , \quad \Gamma \vee \Delta \vee \neg\sigma(V_l) // \Phi\}}$$

if there is a substitution $\sigma$, a position p and a clause $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l // \Psi$ such that one of the following items holds:

- $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l // \Psi \in L$, $\Gamma \equiv \sigma(U_1 \vee \ldots \vee U_k)$, $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l$ is subsumption reductive w.r.t. $\sigma$, $\Delta \vee \Phi \vee \sigma(V_1) \vee \ldots \vee \sigma(V_l)$ is not tautological.
- $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l // \Psi \in H$, $\Gamma \equiv \sigma(U_1 \vee \ldots \vee U_k)$, $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l$ is subsumption reductive w.r.t. $\sigma$, $\Delta \vee \Phi \vee \sigma(V_1) \vee \ldots \vee \sigma(V_l)$ is not tautological and
  (a) $\sigma$ is no permutation or
  (b) $\sigma$ is a permutation and
  (b1) $\Gamma$ is majorized by $\Delta$ or
  (b2) $\Gamma$ is not majorized by $\Delta$ and $\Delta \gg\gg \sigma(V_1) \vee \ldots \vee \sigma(V_l)$ and $(\Delta \vee \Phi) \cap Cpl_<(\sigma(V_1) \vee \ldots \vee \sigma(V_l)) \subseteq \sigma(\Psi)$.

# 8 Correctness and refutational completeness

The proceeding in this section is closely related to [Ba88]. In what we differ from other approaches is the definition of the complexity measure.

We write $(L, H) \vdash (L', H')$ to indicate that $(L', H')$ is the result of applying an inference step on $(L, H)$. The next lemma states (without proof) the correctness of the inference system.

**Lemma 11** *Let* $(L, H) \vdash (L', H')$.
*(a) If* $L \subseteq ITH(T; R)$, *then* $L' \subseteq ITH(T; R)$.
*(b)* $H \subseteq ITH(T; R)$ *iff* $H' \subseteq ITH(T; R)$.
*(c) If the clauses in* $H$ *are majorizing, then the clauses in* $H'$ *are too.*

Complexities are defined for literal and clause instances w.r.t. a model of $T$. A *literal resp. clause instance w.r.t.* $\mathcal{A} \models T$ is a pair $(L, \tau)$ resp. $(\Gamma//\Phi, \tau)$ of a literal resp. clause and a $\Sigma_1(\mathcal{A})$- ground substitution. A *$H$-inconsistency w.r.t.* $\mathcal{A} \models T$ is a clause instance $(\Gamma//\Phi, \tau)$ w.r.t. $\mathcal{A}$ with $\mathcal{T}(\mathcal{A}) \not\models \tau(\Gamma)$ resp. $\tau(\Gamma) \xrightarrow{*}_{R(\mathcal{A})} false$.

We begin by defining the literal complexity $c^L$ (c. [Ba88]): Let $\mathcal{A} \models T$ and $\tau$ be a $\Sigma_1(\mathcal{A})$-ground substitution.

$$c^L(u = v, \tau) = c^L(u \neq v, \tau) = \begin{cases} (\{\tau(u)\}, \{u\}, \tau(v)) & \text{if } u > v \\ (\{\tau(v)\}, \{v\}, \tau(u)) & \text{if } v > u \\ (\{\tau(u), \tau(v)\}, \{u, v\}, -) & \text{otherwise} \end{cases}$$

Let $>^L$ be the lexicographically combination of $\gg$, $\rhd\rhd$ and $>$. We next define three auxiliary complexities $c_1$, $c_2$ and $c_3$. Let $\Gamma//\Phi \equiv C_1 \vee \ldots \vee C_n//\Phi$. Then:

$$\begin{array}{lcl} c_1(\Gamma, \tau) & = & \{c^L(C_i, \tau) \mid i \in \{1, \ldots, n\} \text{ and } C_i \text{ is maximal in } \Gamma\} \\ c_2(\Gamma//\Phi) & = & cpl_<(\Gamma \vee \Phi) \\ c_3(\Gamma, \tau) & = & \{c^L(C_i, \tau) \mid i \in \{1, \ldots, n\}\} \end{array}$$

Each $c_i$ is considered as multiset.
The clausal complexity $c$ is defined as

$$c(\Gamma//\Phi) = \begin{cases} (c_1(\Gamma, \tau), c_2(\Gamma//\Phi), c_3(\Gamma, \tau)) & \text{if } \tau(\Gamma) \xrightarrow{*}_{R(\mathcal{A})} false \\ max & \text{otherwise} \end{cases}$$

where *max* is a new symbol. Let $>^c$ be the lexicographically combination of $>^L>^L$, $\gg$, $>^L>^L$ and let *max* be maximal w.r.t. $>^c$. Note that $>^c$ is noetherian by construction.

The next lemma states the fundamental properties of the complexity measure $>^c$ needed below. As it is rather technical we summarize the ideas. We have to consider two cases:
Case 1: A literal $C$ in $\Gamma$ is exchanged by a clause $\Delta'$ (that is majorized by $C$) to get the new clause $\Gamma'$ and is stored in $\Phi'$ if "necessary".
Case 2: A literal $\neg U$ majorized by $\Gamma$ is added to $\Gamma$ if $\Gamma \vee \Phi \vee U$ is not tautological.
In the first case we have to consider two subcases:
Case 1.1: $C$ is a maximal literal in $\Gamma$.
Case 1.2: $C$ is majorized by $\Gamma \backslash \{C\}$.

In case 1.1 the literal $C$ "contributes" to the complexity measure $c_1$ and consequently $c_1(\Gamma, \tau) >^L>^L c_1(\Gamma', \tau)$.

In case 1.2 we first get $c_1(\Gamma, \tau) \approx c_1(\Gamma', \tau)$ as neither $C$ nor $\Delta'$ "contributes" to $c_1$. As $C$ is stored in the history clause $\Phi'$ we get $c_2(\Gamma//\Phi) \approx c_2(\Gamma'//\Phi')$. Finally we obviously have $c_3(\Gamma, \tau) >^L>^L c_3(\Gamma', \tau)$.

In case 2 $c_1(\Gamma, \tau) \approx c_1(\Gamma \vee \neg U, \tau)$ as $\neg U$ is majorized by $\Gamma$. Now $\neg U$ does not occur in $\Gamma \vee \Phi$ for otherwise $\Gamma \vee \Phi \vee U$ would be tautological. Hence when adding $\neg U$ to $\Gamma$ the $<$-complement properly decreases, i.e. $c_2(\Gamma//\Phi) \gg c_2(\Gamma \vee \neg U//\Phi)$.

**Lemma 12** *Let* $\mathcal{A} \models T$ *and* $\tau$ *be a* $\Sigma_1(\mathcal{A})$*-ground substitution. Let* $\Gamma//\Phi$ *be a clause with* $\tau(\Gamma) \xrightarrow{*}_{R(\mathcal{A})} false$.
*(a) Let* $\Gamma = C \vee \Delta$ *such that* $C$ *is not majorized by* $\Delta$. *Let* $\{C\} \gg\gg \Delta'$ *and* $\tau(\Delta \vee \Delta') \xrightarrow{*}_{R(\mathcal{A})} false$. *Then* $c(\Gamma//\Phi, \tau) >^c c(\Delta \vee \Delta'//\Phi', \tau)$ *for an arbitrary* $\Phi'$.

*(b) Let $U$ be a literal that is majorized by $\Gamma$. Assume that $\Gamma \vee \Phi \vee U$ is not tautological and $\tau(\Gamma \vee \neg U) \xrightarrow{*}_{R(\mathcal{A})}$ false. Then $c(\Gamma//\Phi, \tau) >^c c(\Gamma \vee \neg U//\Phi, \tau)$.*
*(c) Let $\Gamma'//\Phi'$ be a majorizing clause such that $\tau(\Gamma') \xrightarrow{*}_{R(\mathcal{A})}$ false and $\Gamma \gg\gg \Gamma'$ and $(\Gamma \vee \Phi) \cap cpl_<(\Gamma') \subseteq \Phi'$. Then $c(\Gamma//\Phi, \tau) >^c c(\Gamma'//\Phi', \tau)$.*

**Proof:**
(a) We have $c_1(\Gamma, \tau) >^L >^L c_1(\Delta \vee \Delta', \tau)$ as $c^L(D, \tau)$ contributes to $c_1(\Gamma, \tau)$.
(b) As $U$ (resp. $\neg U$) is majorized by $\Gamma$ we get $c_1(\Gamma, \tau) \approx c_1(\Gamma \vee \neg U, \tau)$. $\neg U$ doesn't occur in $\Gamma \vee \Phi$ for otherwise $\Gamma \vee \Phi \vee U$ would be tautological. As $\neg U \in cpl_<(\Gamma \vee \Phi) \backslash cpl_<(\Gamma \vee \neg U \vee \Phi)$ we get $c_2(\Gamma//\Phi) \gg c_2(\Gamma \vee \neg U//\Phi)$.
(c) Obviously $c_1(\Gamma, \tau) >^L >^L c_1(\Gamma', \tau)$ or $c_1(\Gamma, \tau) \approx c_1(\Gamma', \tau)$. We proof $cpl_<(\Gamma' \vee \Phi') \subseteq cpl_<(\Gamma \vee \Phi)$. Let $D \in cpl_<(\Gamma' \vee \Phi')$. Then $D \notin \Gamma' \vee \Phi'$. As $\Gamma'$ majorizes $\Phi'$ there is some $C' \in \Gamma'$ such that $C' \gg D$. As $\Gamma \gg\gg \Gamma'$ there exists some $C \in \Gamma$ such that $C \gg D$. Hence $D$ is majorized by $\Gamma \vee \Phi$. If $D$ were an element of $\Gamma \vee \Phi$ then — as $D \in cpl_<(\Gamma')$ — by assumption $D \in \Phi'$, which contradicts $D \notin \Gamma' \vee \Phi'$. Hence $D \in cpl_<(\Gamma \vee \Phi)$. Consequently $c_2(\Gamma//\Phi) \gg c_2(\Gamma'//\Phi')$ or $c_2(\Gamma//\Phi) \approx c_2(\Gamma'//\Phi')$. Obviously we have $c_3(\Gamma, \tau) >^L >^L c_3(\Gamma', \tau)$. □

**Lemma 13** *Let $(L, H) \vdash (L', H')$. Let $\mathcal{A} \models T$. If $(\Gamma//\Phi, \tau)$ is a $H$-inconsistency w.r.t. $\mathcal{A}$, then either $(\Gamma//\Phi, \tau)$ is a $H'$-inconsistency w.r.t. $\mathcal{A}$ or there exists a $H'$-inconsistency $(\Gamma'//\Phi', \tau')$ w.r.t. $\mathcal{A}$ such that $c(\Gamma//\Phi, \tau) >^c c(\Gamma'//\Phi', \tau')$.*

**Proof:**
The cases (DedHyp), (IntLem), (DelTau), (DelDed) and (DelEqu) are trivial. The cases (SimpEqu) and (SimpElim) are treated with lemma 12(c).
(SimpRew): Let $\tau(c \vee \Gamma) \xrightarrow{*}_{R(\mathcal{A})}$ false resp. $\mathcal{T}(\mathcal{A}) \not\models \tau(C \vee \Gamma)$. ($\mathcal{A}$ and $\tau$ as claimed)
Case 1: simplification with $p$, $\sigma$ and $< u = v > \vee U_1 \vee \ldots \vee U_n \in R \cup L$:
Case 1.1: $\mathcal{T}(\mathcal{A}) \not\models \tau(\sigma(U_1) \vee \ldots \vee \sigma(U_n))$: Then $\mathcal{T}(\mathcal{A}) \models \tau(\sigma(u) = \sigma(v))$. Further $\mathcal{T}(\mathcal{A}) \not\models \tau(C' \vee \Gamma \vee \sigma(U_1) \vee \ldots \vee \sigma(U_n))$ for otherwise $\mathcal{T}(\mathcal{A}) \models \tau(C)$. By lemma 12(c) we get $c(C \vee \Gamma//\Phi, \tau) >^c c(C' \vee \Gamma \vee \sigma(U_1) \vee \ldots \vee \sigma(U_n)//\Phi', \tau)$.
Case 1.2: $\mathcal{T}(\mathcal{A}) \models \tau(\sigma(U_i))$ for some $i$: Then we get the desired result by lemma 12(b).
Case 2: simplification with $p$, $\sigma$ and $< u = v > \vee U_1 \vee \ldots \vee U_n//\Psi \in H$:
Case 2.1: $\mathcal{T}(\mathcal{A}) \not\models \tau(\sigma(U_1) \vee \ldots \vee \sigma(U_n))$:
Case 2.1.1: $\mathcal{T}(\mathcal{A}) \models \tau(\sigma(u = v \vee U_1 \vee \ldots \vee U_n))$: Proceed as in case 1.1.
Case 2.1.2: $\mathcal{T}(\mathcal{A}) \not\models \tau(\sigma(u = v \vee U_1 \vee \ldots \vee U_n))$: As we have $c^L(C, \tau) >^L c^L(u = v, \tau\sigma) >^L c^L(U_i, \tau\sigma)$ we get $c_1(C \vee \Gamma, \tau) >^L >^L c_1(u = v \vee U_1 \vee \ldots \vee U_n, \tau\sigma)$.
Case 2.2: as case 1.2
(SimpSub): Let $\tau(\Gamma \vee \Delta) \xrightarrow{*}_{R(\mathcal{A})}$ false resp. $\mathcal{T}(\mathcal{A}) \not\models \tau(\Gamma \vee \Delta)$.
Case 1: subsumption with $p$, $\sigma$ and $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l \in L$: Then $\mathcal{T}(\mathcal{A}) \models \sigma(V_i)$ for some $i$. By lemma 12(b) we get the desired result.
Case 2: subsumption with $p$, $\sigma$ and $< U_1 \vee \ldots \vee U_k > \vee V_1 \vee \ldots \vee V_l//\Psi \in H$:
Case 2.1: $\mathcal{T}(\mathcal{A}) \not\models \tau(\sigma(V_1) \vee \ldots \vee \sigma(V_l))$: Then $\mathcal{T}(\mathcal{A}) \not\models \tau(\sigma(U_1 \vee \ldots \vee U_k \vee V_1 \vee \ldots \vee V_l))$.
Case 2.1.1: $\sigma$ is no permutation or $\sigma$ is a permutation and $\Gamma$ is majorized by $\Delta$: Then $c_1(\Gamma \vee \Delta, \tau) >^L >^L c_1(U_1 \vee \ldots \vee U_k \vee V_1 \vee \ldots \vee V_l, \tau\sigma)$.
Case 2.1.2: $\sigma$ is a permutation, $\Gamma$ is not majorized by $\Delta$, $\Delta \gg\gg \sigma(V_1) \vee \ldots \vee \sigma(V_l)$ and $(\Delta \vee \Phi) \cap Cpl_<(\sigma(V_1) \vee \ldots \vee \sigma(V_l)) \subseteq \sigma(\Psi)$: By lemma 12(c) we get $c(\Gamma \vee \Delta, \tau) >^c c(\sigma(U_1) \vee \ldots \vee \sigma(U_k) \vee \sigma(V_1) \vee \ldots \vee \sigma(V_l), \tau)$.
Case 2.2: analogous to case 1. □

For a (modulo $T$) refutational complete theorem prover we need "fair" derivations:

**Definition 11**
*(a) $H'$ is a* $T$-covering set for $H$ (w.r.t. $R$) *iff for any $\mathcal{A} \models T$ and each $H$-inconsistency $(\Gamma//\Phi, \tau)$ w.r.t. $\mathcal{A}$ there exists a $H \cup H'$-inconsistency $(\Gamma'//\Phi', \tau')$ w.r.t. $\mathcal{A}$ such that $c(\Gamma//\Phi, \tau) >^c c(\Gamma'//\Phi', \tau')$.*
*(b) A derivation $(L_0, H_0) \vdash (L_1, H_1) \vdash \cdots$ is said to be* fair *iff the set $\bigcup_{i \in \mathbf{N}} H_i$ of all deduced clauses is a $T$-covering set for the set $\bigcup_{i \in \mathbf{N}} \bigcap_{j \geq i} H_j$ of all* persisting *clauses.*

**Theorem 4** *Let $(L_0, H_0) \vdash (L_1, H_1) \vdash \cdots$ be a fair derivation.*
*(a) If $H_0$ is not inductively valid w.r.t $R$ and $T$, then there exists an $i \in \mathbf{N}$ such that $H_i$ contains a $\Sigma_0$-clause*

*that is not parameter valid w.r.t. $T$. ("refutational completeness mod. $T$")*
*(b) If $\bigcup_{i\leq k} H_i$ is a $T$-covering set for $H_k$ and if every $\Sigma_0$-clause in $\bigcup_{i\leq k} H_i$ is parameter valid w.r.t. $T$, then $\bigcup_{i\leq k} H_i \subseteq ITH(T;R)$.*

**Proof:** as in [Ba88]

# 9 $T$-covering sets

Theorem 4 in the last section requires a fair derivation to obtain a (modulo $T$) refutational complete theorem prover. In this section we give a sufficient condition under which one always can produce a fair derivation.

We want to guarantee that a clause $\Gamma//\Phi$ that cannot be treated by a built-in algorithm — i.e. that contains a new symbol — can be covered by a set of critical clauses between $R$ and $\Gamma$ at a maximal literal $C$ in $\Gamma$ as indicated in the formulation of the inference system. Two problems arise: We firstly want to guarantee that an instantiated clause containing a new symbol is $\longrightarrow_{R(\mathcal{A})}$-reducible by some equation from $R$ and secondly that such a reduction is possible at a maximal literal in the clause (such that this literal contributes to the complexity measure $c_1$).

To get the first property we require $R$ to be parameter leftlinear. We write $\longrightarrow_{R(\mathcal{A});R}$ resp. $\longrightarrow_{R(\mathcal{A});\mathcal{A}}$ to indicate that an equation from $R$ resp. $DIA^+(\mathcal{A})$ is used in the rewrite step. The next lemma states (without proof) the interchangeability of the rewrite steps using $R$ and $DIA^+(\mathcal{A})$.

**Lemma 14** *Let $R$ be parameter simple and parameter leftlinear. Let $R$ be $T$-ground confluent (as usual). Let $\mathcal{A} \models T$. If $s \xrightarrow{*}_{R(\mathcal{A});\mathcal{A}} t \longrightarrow_{R(\mathcal{A});R} u$ for $s,t,u \in TERM(F_1(\mathcal{A}))$, then there exists a term $v \in TERM(F_1(\mathcal{A}))$ such that $s \longrightarrow_{R(\mathcal{A});R} v \xrightarrow{*}_{R(\mathcal{A});\mathcal{A}} u$.*

In order to guarantee that a clause containing a new symbol contains a maximal literal with a new symbol we require $>$ to be compatible with the specification hierarchy.

**Definition 12** *An ordering $>$ is said to be* compatible with the specification hierarchy *iff no term $s \in TERM(F_0,V_0)$ with only parameter symbols is greater w.r.t. $>$ than any term $t \in TERM(F_1,V_1)$ containing a new symbol from $F_1\backslash F_0$.*

The next lemma now assures the existence of fair derivations. Consequently, if the premises of the lemma are fulfilled , we can construct a modulo $T$ refutational theorem prover. The way how such a prover works will then be described in the next section.

**Lemma 15** *Let $R$ be $T$-ground confuent and $T$-sufficient complete w.r.t. parameter sorts (as usual). Let $R$ further be parameter simple and parameter leftlinear. Let $>$ be compatible with the specification hierarchy. Let $\Gamma//\Phi$ be a clause containing a new symbol from $F_1\backslash F_0$. Then there exists a maximal literal $C_i$ in $\Gamma \equiv C_1 \vee \ldots \vee C_k$ such that the set $CRITCLAUSE(R,\Gamma,C_i)$ of critical clauses between $R$ and $\Gamma$ at the literal $C_i$ is a $T$-covering set for $\{\Gamma//\Phi\}$.*

**Proof:** Let $\mathcal{A} \models T$ and $\tau$ be a $\Sigma_1(\mathcal{A})$-ground substitution with $\tau(\Gamma) \xrightarrow{*}_{R(\mathcal{A})} false$. As $>$ is compatible with the specification hierarchy there is some literal $C_i$ in $\Gamma$ that contains a new symbol and that is not majorized by $\Gamma\backslash\{C_i\}$. By $T$-sufficient completeness (and our overall assumption of clauses to be over sort $S_0$) we know that $\tau(C_i)$ is $\longrightarrow_{R(\mathcal{A})}$-reducible. By the previous lemma we know that $\tau(C_i)$ is even $\longrightarrow_{R(\mathcal{A});R}$-reducible.
Case 1: There exists a variable $x$ in $C_i$ such that $\tau(x)$ is $\longrightarrow_{R(\mathcal{A});R}$-reducible. Consider $\tau'$ with $\tau'(x)$ being the $\longrightarrow_{R(\mathcal{A});R}$-normal form of $\tau(x)$. Then $(\Gamma//\Phi,\tau')$ has the desired property.
Case 2: There is no variable $x$ in $C_i$ such that $\tau(x)$ is $\longrightarrow_{R(\mathcal{A});R}$-reducible. Then there exists a conditional equation $< u = v > \vee U \in R$ $(U \equiv U_1 \vee \ldots \vee U_n)$ and a critical clause $\mu(C_1) \vee \ldots \vee \mu(C_i)[p \leftarrow \mu(v)] \vee \ldots \vee \mu(C_k) \vee \mu(U)//\Phi' \in CRITCLAUSE(R,\Gamma,C_i)$ and a substitution $\tau'$ such that $\tau(\neg U) \xrightarrow{*}_{R(\mathcal{A})} true$ and $\tau = \tau'\mu$. Hence $\tau'(\mu(C_1) \vee \ldots \vee \mu(C_i)[p \leftarrow \mu(v)] \vee \ldots \vee \mu(C_k) \vee \mu(U)) \xrightarrow{*}_{R(\mathcal{A})} false$. By lemma 12(a) we get $c(\Gamma//\Phi,\tau) >^c c(\mu(C_1) \vee \ldots \vee \mu(C_i)[p \leftarrow \mu(v)] \vee \ldots \vee \mu(C_k) \vee \mu(U)//\Phi',\tau')$. □

The question arizes of how restrictive parameter leftlinearity is. The following lemma and the succeeding corollary state (without proof) that it is not at all in our context.

**Lemma 16** *Let $R$ be parameter simple, $T$-ground confluent and compatible with a $T$-extendable reduction ordering $>$ (as usual). Let $u = v$ if $U \in R$ be a conditional equation with two distinct positions $q_1, q_2 \in O(u)$ such that $u/q_1 \equiv u/q_2 \equiv x \in V_0$. Let $y$ be a new variable and $R'$ the rewrite system that results from $R$ where $(u = v$ if $U)$ is exchanged by $(u[q_2 \leftarrow y] = v$ if $U \wedge x = y)$. Then for any $\mathcal{A} \models T$ and $s, t \in TERM(F_1(\mathcal{A}))$ we have:*
*(a)* $s \overset{*}{\longleftrightarrow}_{R(\mathcal{A})} t$ *iff* $s \overset{*}{\longleftrightarrow}_{R'(\mathcal{A})} t$
*(b) $R'$ is $T$-ground confluent.*
*(c) $R$ and $R'$ produce the same equivalences on $TERM(F_1(\mathcal{A}))$, i.e.:* $\overset{*}{\longleftrightarrow}_{G(\mathcal{A})} \equiv \overset{*}{\longleftrightarrow}_{G'(\mathcal{A})}$.

**Corollary 4** *Let $R$ be parameter simple, $T$-ground confluent and compatible with a $T$-extendable reduction ordering $>$. Then there exists a $T$-ground confluent $R'$ (compatible with $>$) that is parameter leftlinear and produces the same equivalences on $TERM(F_1(\mathcal{A}))$ for any $\mathcal{A} \models T$.*

# 10 The Theorem Prover

We first describe how the theorem prover works and then demonstrate it by considering an example.

The prover repeatedly performs the following (conditional) operations: (note that at the beginning all clauses are assumed to be unmarked)

(1) All clauses in $H$ are marked: Accept the hypotheses.
(2) There exist unmarked clauses in $H$:
(2.1) There are no $\Sigma_0$-clauses in $H$:
(2.1.1) No simplification and deletion rule is appliable: Mark a (unmarked) clause and cover it.
(2.1.2) A simplification or deletion rule is appliable: Apply it.
(2.2) There are $\Sigma_0$-clauses in $H$:
(2.2.1) All $\Sigma_0$-clauses are parameter valid w.r.t. $T$: Delete them with *DelDed*.
(2.2.2) There is a $\Sigma_0$-clause that is not parameter valid w.r.t. $T$: Stop and refute the hypotheses.

To illustrate the prover we return to example 1 introduced in section 2 where $T$ consists of the axioms of totally ordered sets and in addition of two standard boolean axioms. We want to prove that the clauses (A) ,..., (E) are inductivley valid w.r.t. $R$ and $T$. As we only want to demonstrate the method and not to present all details here we only consider the proofs of (A), (B) and (E) (using (A) ,..., (D) as lemmata). When applying *DelDed* (i.e. the built-in algorithm) we use in this section an oracle. In the next section we present an algorithm to replace the oracle.

As ordering $>$ we may use a semantical recursive path ordering as in [Gr90] (that can be chosen to be compatible with the specification hierarchy and to be $T$-extendable). The system $R$ of conditional equations is parameter simple and parameter leftlinear. One easily proves that $R$ is $T$-ground confluent and $T$-sufficient complete w.r.t. parameter sorts.

To shorten the derivations we use in addition a simplification rule (SimpRew'). Note that every step using this rule can be replaced by succesive (SimpRew)-steps.

(SimpRew' — simplification by clausal rewriting)

$$\frac{L, H \cup \{C \vee \Gamma // \Phi\}}{L, H \cup \{C'_1 \vee \Gamma \vee \sigma(U_1) // \Phi'_1 \quad , \ldots , \quad C'_n \vee \Gamma \vee \sigma(U_n) // \Phi'_n\}}$$

if there exist a substitution $\sigma$, a position p and clauses $< u = v_1 > \vee U_1 \, , \ldots , < u = v_n > \vee U_n \in R$ so that: $C/p \equiv \sigma(u)$, $C'_i \equiv C[p \leftarrow \sigma(v_i)]$ $(i = 1 \ldots n)$, $\Phi'_i \equiv (\Phi \cup \{C\}) \cap cpl_<(C'_i \vee \Gamma)$ $(i = 1 \ldots n)$ and $\neg U_1 \vee \ldots \vee \neg U_n$ is logically valid.

Below we have listed derivations that prove the clauses (A), (B) and (E). The information in the right column indicates the rule that is applied next to the clause in the middle column. The item in the left column indicates the results that are obtained by applying a rule : for instance by applying (DedHyp) on the clause with item (A) we get the clauses with items (A.1), (A.2) and (A.3). The additional star indicates the marked (covered) clauses. History clauses are not needed here as always maximal literals are treated.

| | | | |
|---|---|---|---|
| (*) | $(A)$ | $(min(a, A) > a \neq t)$ | *DedHyp* |
| | $(A.1)$ | $(a > a \neq t)$ | *DelDed* |
| (*) | $(A.2)$ | $(min(b, B) > a \neq t) \vee (a > b \neq t)$ | *DedHyp* |
| | $(A.2.1)$ | $(b > a \neq t) \vee (a > b \neq t)$ | *DelDed* |
| | $(A.2.2)$ | $(min(c, C) > a \neq t) \vee (a > b \neq t) \vee (b > c \neq t)$ | *SimpSub*$(A)$ |
| | $(A.2.2)$ | $\top$ | *DelTau* |
| (*) | $(A.2.3)$ | $(min(b, C) > a \neq t) \vee (a > b \neq t) \vee (b > c \neq t)$ | *DedHyp* |
| | $(A.2.3.1)$ | $(b > a \neq t) \vee (a > b \neq t) \vee (b > c = t)$ | *DelDed* |
| | $(A.2.3.2)$ | $(min(d, D) > a \neq t) \vee (a > b \neq t) \vee (b > c = t) \vee (b > d \neq t)$ | *SimpSub*$(A.2)$ |
| | $(A.2.3.2.1)$ | $\top$ | *DelTau* |
| | $(A.2.3.2.2)$ | $(a > b \neq t) \vee (b > c = t) \vee (b > d \neq t) \vee (b > d = t)$ | *DelDed* |
| | $(A.2.3.3)$ | $(min(b, D) > a \neq t) \vee (a > b \neq t) \vee (b > c = t) \vee (b > d = t)$ | *SimpSub*$(A.2.3)$ |
| | $(A.2.3.3)$ | $\top$ | *DelTau* |
| | $(A.3)$ | $(min(a, B) > a \neq t) \vee (a > b = t)$ | *SimpSub*$(A)$ |
| | $(A.3)$ | $\top$ | *DelTau* |
| (*) | $(B)$ | $(leq(min(x, A), A) = t)$ | *DedHyp* |
| | $(B.1)$ | $(leq(x, nil) = t)$ | *SimpRew* |
| | $(B.1)$ | $(t = t)$ | *DelTau* |
| | $(B.2)$ | $(leq(min(a, A), a.A) = t) \vee (x > a \neq t)$ | *SimpRew'* |
| | $(B.2.1)$ | $(f = t) \vee (x > a \neq t)) \vee (min(a, A) > a \neq t)$ | *SimpElim* |
| | $(B.2.1)$ | $(x > a \neq t)) \vee (min(a, A) > a \neq t)$ | *SimpSub*$(A)$ |
| | $(B.2.1)$ | $\top$ | *DelTau* |
| | $(B.2.2)$ | $(leq(min(a, A), A) = t) \vee (x > a \neq t) \vee (min(a, A) > a = t)$ | *SimpSub*$(B)$ |
| | $(B.2.2)$ | $\top$ | *DelTau* |
| | $(B.3)$ | $(leq(min(x, A), a.A) = t) \vee (x > a = t)$ | *SimpRew'* |
| | $(B.3.1)$ | $(f = t) \vee (x > a = t) \vee (min(x, A) > a \neq t)$ | *SimpElim* |
| (*) | $(B.3.1)$ | $(x > a = t) \vee (min(x, A) > a \neq t)$ | *DedHyp* |
| | $(B.3.1.1)$ | $(x > a = t) \vee (x > a \neq t)$ | *DelTau* |
| | $(B.3.1.2)$ | $(x > a = t) \vee (min(b, B) > a \neq t) \vee (x > b \neq t)$ | *SimpSub*$(B.3.1)$ |
| | $(B.3.1.2.1)$ | $\top$ | *DelTau* |
| | $(B.3.1.2.2)$ | $(x > a = t) \vee (x > b \neq t) \vee (b > a \neq t)$ | *DelDed* |
| | $(B.3.1.3)$ | $(x > a = t) \vee (min(x, B) > a \neq t) \vee (x > b = t)$ | *SimpSub*$(B.3.1)$ |
| | $(B.3.1.3)$ | $\top$ | *DelTau* |
| | $(B.3.2)$ | $(leq(min(x, A), A) = t) \vee (x > a = t) \vee (min(x, A) > a = t)$ | *SimpSub*$(B)$ |
| | $(B.3.2)$ | $\top$ | *DelTau* |
| (*) | $(E)$ | $(ord(sort(A)) = t)$ | *DedHyp* |
| | $(E.1)$ | $(ord(nil) = t)$ | *SimpRew* |
| | $(E.1)$ | $(t = t)$ | *DelTau* |
| | $(E.2)$ | $(ord(min(b, B).sort(del(min(b, B), b.B))) = t)$ | *SimpRew'* |
| | $(E.2.1)$ | $(ord(sort(del(min(b, B), b.B))) = t) \vee (leq(\ldots) \neq t)$ | *SimpRew*$(E)$ |
| | $(E.2.1)$ | $(t = t) \vee (leq(\ldots) \neq t)$ | *DelTau* |
| | $(E.2.2)$ | $(f = t) \vee (leq(min(b, B), sort(del(min(b, B), b.B))) = t)$ | *SimpElim* |
| | $(E.2.2)$ | $(leq(min(b, B), sort(del(min(b, B), b.B))) = t)$ | *SimpRew*$(D)$ |
| | $(E.2.2)$ | $(leq(min(b, B), del(min(b, B), b.B)) = t)$ | *SimpRew*$(C)$ |
| | $(E.2.2.1)$ | $(leq(min(b, B), b.B) = t) \vee (min(b, B) > min(b, B) = t)$ | *SimpElim* |
| | $(E.2.2.1)$ | $(leq(min(b, B), b.B) = t)$ | *SimpRew'* |
| | $(E.2.2.1.1)$ | $(f = t) \vee (min(b, B) > b \neq t)$ | *SimpSub*$(A)$ |
| | $(E.2.2.1.1)$ | $(f = t) \vee \top$ | *DelTau* |
| | $(E.2.2.1.2)$ | $(leq(min(b, B), B) = t) \vee (min(b, B) > b = t)$ | *SimpSub*$(B)$ |
| | $(E.2.2.1.2)$ | $\top \vee (min(b, B) > b = t)$ | *DelTau* |
| | $(E.2.2.2)$ | $(leq(\ldots) = t) \vee (min(b, B) > min(b, B) \neq t)$ | *DelSub* |

# 11 A Built-in Algorithm

Finally we present a procedure to decide whether a $\Sigma_0$-clause $\Gamma$ is a logical consequence of $T$ or — equivalently — whether $T \cup \{\neg\Gamma\}$ is not satisfiable. We consider the special case of example 1. Hence

$$S_0 = \{boole, element\}$$
$$F_0 = \{t, f, >\}$$

$$\begin{aligned} T_{boole} &= \{t \neq f\ ,\ \forall x : boole.\ x = t \vee x = f\} \\ T_{totordset} &= \{\forall x, y, z : element.\ (x > x \neq t) \wedge \\ &\qquad (x > y = t\ \wedge\ y > z = t\ \Rightarrow\ x > z = t) \wedge \\ &\qquad (x > y = t\ \vee\ x = y\ \vee\ y > x = t)\} \end{aligned}$$

No variables of type *boole* occur in the example. So we consider only the case of clauses resp. literals with variables ranging over the sort *element*.

Let $\Gamma \in CLAUSE(F_0, V_{element})$. Then $\neg\Gamma$ is a conjunction of literals from $LIT(F_0, V_{element})$. Every literal has one of the following forms (possibly interchanging the left and right hand sides of the equations): $x = y, x \neq y, x > y = t, x > y \neq t, x > y = f, x > y \neq f, t = t, t \neq t, t = f, t \neq f, f = f, f \neq f$.
We will transfer $\neg\Gamma$ with the inference system to follow until no rule application is possible. Then one is able to decide whether $T \cup \{\neg\Gamma\}$ is satisfiable or not. Let $A$ be a set of atomic formulas over $F_0$ and $V_{element}$. Let $\bot$ be an extra literal that is unsatisfiable in any algebra.

$$\frac{A \cup \{L_1 \wedge \ldots \wedge L_i \wedge \ldots \wedge L_n\}}{A \cup \{\bot\}} \text{ if } L_i \in \{x \neq x, x > x = t, x > x \neq f, t \neq t, t = f, f \neq f\}$$

$$\frac{A \cup \{L_1 \wedge \ldots \wedge L_i \wedge \ldots \wedge L_n\}}{A \cup \{L_1 \wedge \ldots \wedge L_{i-1} \wedge \top \wedge L_{i+1} \wedge \ldots \wedge L_n\}} \text{ if } L_i \in \{x = x, x > x \neq t, x > x = f, t = t, t \neq f, f = f\}$$

$$\frac{A \cup \{L_1 \wedge \ldots \wedge (x \neq y) \wedge \ldots \wedge L_n\}}{A \cup \{L_1 \wedge \ldots \wedge (x > y = t \vee y > x = t) \wedge \ldots \wedge L_n\}} \text{ if } x \not\equiv y$$

$$\frac{A \cup \{L_1 \wedge \ldots \wedge L_i \wedge \ldots \wedge L_n\}}{A \cup \{L_1 \wedge \ldots \wedge (y > x = t \vee x = y) \wedge \ldots \wedge L_n\}} \text{ if } x \not\equiv y \text{ and } L_i \in \{x > y \neq t, x > y = f\}$$

$$\frac{A \cup \{L_1 \wedge \ldots \wedge (x > y \neq f) \wedge \ldots \wedge L_n\}}{A \cup \{L_1 \wedge \ldots \wedge (x > y = t) \wedge \ldots \wedge L_n\}}$$

$$\frac{A \cup \{L_1 \wedge \ldots \wedge (L_{i_1} \vee L_{i_2}) \wedge \ldots \wedge L_n\}}{A \cup \{L_1 \wedge \ldots \wedge L_{i_1} \wedge \ldots \wedge L_n\ ,\ L_1 \wedge \ldots \wedge L_{i_2} \wedge \ldots \wedge L_n\}}$$

$$\frac{A \cup \{L\}}{A \cup \{L \wedge (x = z)\}} \text{ if } (x = z) \notin L \text{ and for some } y\text{: } (x = y), (y = z) \in L$$

$$\frac{A \cup \{L\}}{A \cup \{L \wedge (x > z = t)\}} \text{ if } (x > z = t) \notin L \text{ and for some } y\text{: } (x > y = t), (y > z = t) \in L$$

$$\frac{A \cup \{L\}}{A \cup \{L \wedge (x > z = t)\}} \text{ if } (x > z = t) \notin L \text{ and for some } y\text{: } (x > y = t), (y = z) \in L$$

$$\frac{A \cup \{L\}}{A \cup \{L \wedge (z > y = t)\}} \text{ if } (z > y = t) \notin L \text{ and for some } x\text{: } (x > y = t), (x = z) \in L$$

We write $A \rightsquigarrow A'$ if $A'$ is the result of applying an inference rule on $A$. A conjunction $L = L_1 \wedge \ldots \wedge L_n$ of literals is said to be *closed w.r.t. consequences* iff no one of the last four rules is appliable on $L$.

The following lemma summarizes some easy to prove results concerning the transformation of sets of atomic formulas by the inference system. We only scetch the proof of part (c).

**Lemma 17**
*(a) If $A$ is a finite set of atomic formulas (over $F_0$ and $V_{element}$) then there exists a set $A'$ of atomic formulas with $A \overset{*}{\rightsquigarrow} A'$ that cannot be transformed any more. (i.e.: $\rightsquigarrow$ is terminating.)*
*(b) Let $A \overset{*}{\rightsquigarrow} A'$ such that $A'$ cannot be transformed any more. Then $L \in A'$ iff $L \equiv \bot$ or $L \equiv L_1 \wedge \ldots \wedge L_n$ such that $L$ is closed w.r.t. consequences and $L_i \equiv \top$ or $L_i \equiv (x = y)$ or $L_i \equiv (x > y = t)$ where $x$ and $y$ are distinct variables $(i = 1, \ldots, n)$.*
*(c) If $L \equiv L_1 \wedge \ldots \wedge L_n$ such that $L$ is closed w.r.t. consequences and $L_i \equiv \top$ or $L_i \equiv (x = y)$ or $L_i \equiv (x > y = t)$ where $x$ and $y$ are distinct variables $(i = 1, \ldots, n)$, then $T \cup \{L\}$ is satisfiable.*
*(d) Let $A \overset{*}{\rightsquigarrow} A'$. Then $T \cup \{L\}$ is satisfiable for some $L \in A$ iff $T \cup \{L'\}$ is satisfiable for some $L' \in A'$.*

**Proof:** (c) Let $L \equiv L_1 \wedge \ldots \wedge L_n$ such that $L$ is closed w.r.t. consequences and $L_i \equiv \top$ or $L_i \equiv (x = y)$ or $L_i \equiv (x > y = t)$ where $x$ and $y$ are distinct variables $(i = 1, \ldots, n)$. Let $\mathcal{R}_{boole}$ be the $\Sigma_0$-algebra with the set of real numbers as carrier of sort *element*, with a standard boolean part and with $>^{\mathcal{R}}$ being the natural interpretation of $>$. Note that $\mathcal{R}_{boole}$ is a model of $T$. We will construct an assignment $\beta$ of the variables — the construction being presented in form of an algorithm — such that $\mathcal{R}_{boole}$ satisfies $L$ w.r.t. this assignment.

BEGIN
$X := var(L)$
$Y := \emptyset$
WHILE $X \neq \emptyset$ DO
  BEGIN
  let $x$ be an element of $X$
  $Y_< := \{y \in Y \mid (x > y = t) \in L\}$
  $Y_= := \{y \in Y \mid (x = y) \in L\}$
  $Y_> := \{y \in Y \mid (y > x = t) \in L\}$
  IF $Y_= \neq \emptyset$ THEN $\beta(x) := \beta(y)$ for some $y \in Y_=$ ELSE
    BEGIN
    IF $Y_< = \emptyset$ THEN $val_< := 0$ ELSE $val_< := max\{\beta(y) \mid y \in Y_<\}$
    IF $Y_> = \emptyset$ THEN $val_> := 1$ ELSE $val_> := min\{\beta(y) \mid y \in Y_>\}$
    $\beta(x) := (val_< + val_>)/2$
    END
  $X := X \backslash \{x\}$
  $Y := Y \cup \{x\}$
  END
END

$X$ denotes the set of variables that are not assigned a value and $Y$ the set of variables that have been assigned a value. One easily proves that the following statements are invariants of the loop:

- If $(x = y) \in L$ and $x, y \in Y$ then $\beta(x) = \beta(y)$.
- If $(x > y = t) \in L$ and $x, y \in Y$ then $\beta(x) >^{\mathcal{R}} \beta(y) = t^{\mathcal{R}}$.

As the loop obviously terminates we get the desired result. □

**Corollary 5** *Let $S_0 = \{boole, element\}$ and $F_0 = \{t, f, >\}$. Let $T = T_{boole} \cup T_{totordset}$. Then for every clause $\Gamma \in CLAUSE(F_0, V_{element})$ it is decidable whether $T \cup \{\neg\Gamma\}$ is satisfiable resp. whether $\Gamma$ is a logical consequence of $T$.*

**Acknowledgements:** I would like to thank J. Avenhaus for many valuable discussions and helpful suggestions.

# References


[AvMa90] J. Avenhaus and K. Madlener, Term Rewriting and Equational Reasoning, in: R. B. Banerji, ed., *Formal Techniques in Artificial Intelligence* (North-Holland, 1990) pp. 1-43.

[Ba88] L. Bachmair, Proof by Consistency in Equational Theories, in: *3rd LICS* (1988) pp. 228-233.

[BaGa91] L. Bachmair and H. Ganzinger, Perfect Model Semantics for Logic Programs with Equality, in: *Proc. of 8th Int. Conf. on Logic Programming* (MIT Press 1991) pp. 645-659.

[BeLe91] E. Bevers and J. Lewi, Proof by Consistency in Conditional Equational Theories, in: *Conditional and Typed Rewriting Systems — 2nd CTRS '90*, LNCS 516 (Springer, 1991) pp. 194-205.

[BoMo79] R. S. Boyer and J. S. Moore, *A Computational Logic* (Academic Press, 1979).

[Bu69] R. Burstall, Proving Properties of Programs by Structural Induction, *Computer Journal* 12 (1969) pp. 41-48.

[De87] N. Dershowitz, Termination of Rewriting, *J. Symbolic Computation* 3 (1987) pp. 69-116.

[DeJo90] N. Dershowitz and J. P. Jouannaud, Rewriting Systems, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (Elsevier, 1990) pp. 241-320.

[DeOkSi88] N. Dershowitz, M. Okada and G. Sivakumar, Canonical Conditional Rewrite Systems, in *9th International Conference on Automated Deduction*, LNCS 310, (Springer, 1988) pp. 538-549.

[Ga87] H. Ganzinger, Ground Term Confluence in Parametric Conditional Equational Specifications, in: *STACS '87*, LNCS 247 (Springer, 1987) pp. 286-298.

[GoMe87] J. Goguen and J. Meseguer, Models and Equality for Logic Programming, in *TAPSOFT '87*, LNCS 250, (Berlin, 1987) pp. 1-22.

[Gr90] B. Gramlich, Completion Based Inductive Theorem Proving — A Case Study in Verifying Sorting Algorithms, SEKI Report SR-90-04.

[HuHu82] G. Huet and J. M. Hullot, Proofs by Induction in Equational Theories with Constructors, *J. Comput. Syst. Sci* 25 (1982) pp. 239-266.

[Ka88] S. Kaplan, Positive/Negative Conditional Rewriting, in: *Conditional Term Rewriting Systems*, LNCS 308 (Springer 1987) pp. 129-143.

[KaCh86] S. Kaplan and M. Choquer, On the Decidability of Quasi-Reducability, in: *Bull. EATCS 28* (1986) pp. 32-34.

[Ki91] H. Kirchner, Proofs in Parametrized Specifications, in: *4th RTA '91*, LNCS 488 (Springer, 1991) pp. 174-187.

[KoRu90] E. Kounalis and M. Rusinowitch, Mechanizing Inductive Reasoning, in: *Proc. of 8th AAAI '90* (MIT Press, 1990) pp. 240-245.

[Mu80] D. R. Musser, On Proving Inductive Properties of Abstract Data Types, in: *Proc. 7th ACM Symp. on Principles of Programming Languages* (1980) pp. 154-162.

[NiOr91] R. Nieuwenhuis and F. Orejas, Clausal Rewriting, in: *Conditional and Typed Rewriting Systems — 2nd CTRS '90*, LNCS 516 (Springer, 1991) pp. 246-261.

[ZaRe85] H. Zang and J. L. Rémy, Contextual Rewriting, in: *1st RTA '85*, LNCS 202 (Springer, 1985) pp. 46-62.

[Wi91] C.-P. Wirth, Inductive Theorem Proving in Theories specified by Positive/Negative Conditional Equations, Diplomarbeit, Universität Kaiserslautern, Fachbereich Informatik, 1991.