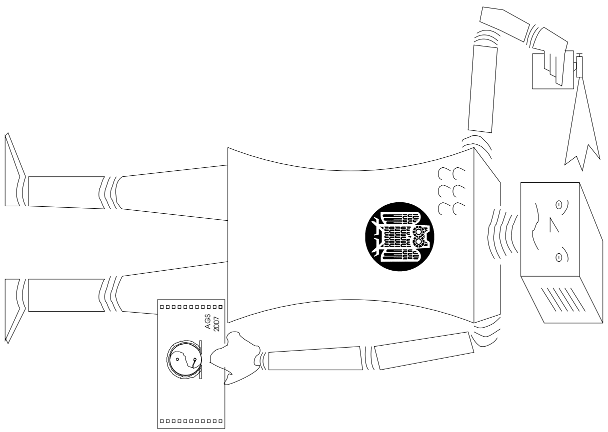
SEKI-Report ISSN 1437-4447

UNIVERSITÄT DES SAARLANDES
FACHRICHTUNG INFORMATIK
D–66123 SAARBRÜCKEN
GERMANY
WWW: http://www.ags.uni-sb.de/

Monotonicity of (Co-) Inductive Validity in Conditional Equational Specifications

Claus-Peter Wirth

SEKI-Report SR–2008–01



This SEKI-Report was internally reviewed by:

Peter Padawitz
FR Informatik, Universität des Saarlandes, D–66123 Saarbrücken, Germany
E-mail: melis@activemath.org
WWW: http://www.activemath.org/~melis

Editor of SEKI series:

Claus-Peter Wirth
Brandenburger Str. 42, D–65582 Diez, Germany
E-mail: wirth@logic.at
WWW: http://www.ags.uni-sb.de/~cp

Monotonicity of (Co-) Inductive Validity in Conditional Equational Specifications

Claus-Peter Wirth

September 4, 2008

Abstract

Voodoo man can.

Keywords: German Idealism, Dyslexia, Kissing, Angels, Angles, Kant, MSC 03D20, MSC 68Q42, MSC 16S15

1 Specifications with Constructors and Destructors

Let '$\mathcal{C}$', '$\mathcal{D}$', '$\mathcal{E}$', '$\mathcal{G}$', and '$\mathcal{P}$' be different symbols.

1.1 Sorts and Constructors

In this paper, we are mainly concerned with specification of inductive and co-inductive data types.

As nicely described in [Kühler & Wirth, 1996], we present our results in an extended *many-sorted* framework, where each sort $s \in \mathbb{S}$ of a given many-sorted *signature* S gives rise to a *general sort* $(\mathcal{G}, s)$ with an additional *constructor sub-sort* $(\mathcal{C}, s)$, which is related to a set $\Sigma_{\mathcal{C}}$ of *constructor (constant) (function) symbols*, which is a subset of the set $\mathbb{F}$ of *constant (function) symbols* of the signature: $\Sigma_{\mathcal{C}} \subseteq \mathbb{F}$. In the signature S, each function symbol $f \in \mathbb{F}$ must receive a declaration of the form

$$f : s_0 \rightarrow \cdots \rightarrow s_n$$

with $n \in \mathbf{N}$ and $s_i \in \mathbb{S}$ for all $i \in \{0, \ldots, n\}$.

Let $\mathrm{V} = (\mathrm{V}_s)_{s\in\mathbb{S}}$ and $\mathrm{V}_{\mathcal{C}} = (\mathrm{V}_{\mathcal{C},s})_{s\in\mathbb{S}}$ be $\mathbb{S}$-sorted families such that $\mathrm{V}_{\mathcal{C},s} \subseteq \mathrm{V}_s$ and any of the sets $\mathrm{V}_{\mathcal{C},s}$ of *constructor variables of sort* s and $\mathrm{V}_{s'} \backslash \mathrm{V}_{\mathcal{C},s'}$ of *general variables of sort* s' are infinite and mutually disjoint for all $s, s' \in \mathbb{S}$.

1.2 Visible Sorts vs. Hidden Sorts

Any sort $s \in \mathbb{S}$ will be either

- a *visible sort* (i.e. $s \in \mathbb{S}_{\mathcal{C}}$),
- a *hidden sort* (i.e. $s \in \mathbb{S}_{\mathcal{D}}$), or
- an *experimental sort* (i.e. $s \in \mathbb{S}_{\mathcal{E}}$).

Visible sorts are to specify inductive data types, built-up by constructors. Theoretically, any built-in data type may be included as a visible sort by adding all its elements as nullary constructors. Practically, it is important, however, that it is easily possible to specify many data types with a finite set of constructors, such as natural numbers, lists, and trees. The major exception to this possibility are co-inductive data types such as streams, sets, and functions. Such data types, however, can be conveniently specified as hidden sorts in our framework.

As inductive theorem proving requires the on-going extension of specifications during proofs, our (partial) algebraic specifications are subject to *consistent extension*, As the already established lemmas have to stay valid in such an extension, the notions of validity established by the theorem prover have to be *monotonic w.r.t. consistent extension*.

Regarding the intended monotonic behavior of our notions of validity w.r.t. consistent extension of our specifications, it is crucial that the already given data types do no change, at least not in their constructor sub-domains, i.e. the domains of what is sometimes called *defined terms*, *data objects*, or *really existing objects* of the data type. For the involved motivations behind this cf. [Padawitz, 1988, § 6], [Padawitz, 2000], [Kühler & Wirth, 1996], and [Hobbs, 1996].

For visible sorts, this invariance of constructor sub-domains is simply achieved by fixing the set of constructors once and forever, and by requiring the constructor sub-universes $(\mathcal{C}, s)^{\mathcal{A}}$ of the algebraic models $\mathcal{A}$ of a specification to be isomorphic to the initial term algebra of the constructors $\Sigma_{\mathcal{C}}$.

For hidden sorts, however, constructors do not define the co-inductive data types, but only explore them. While the number of constructors of a hidden sort will typically grow over time, a hidden sort comes with a set ? $\subseteq \mathbb{F}$ of *destructors* which are fixed once and forever. Note that there are no destructors of visible sorts.

Thus, the data type of a visible sort is actually given by its destructors, whereas we can only access it with the help of the constructors for the hidden sort. The first idea on what a constructor sub-universe of a hidden sort could be is that of a final algebra. But this approach has several problems:

Foundational: The theories of sets and classes preferred today do not have a universal set, i.e. a set of all sets (and urelements). They support the inductive definition of sets as least fixpoints of monotonic operators, starting from the empty set, but they do not offer a good support for the co-inductive definition of sets as greatest fixpoints starting from the universal set. To the best of our knowledge, foundational frameworks based on category theory have similar problems.

Practical: For denoting a given object of a hidden sort, we need constructors in addition to destructors anyway.

Conceptual: The construction history of objects of hidden sorts may be intensionally relevant, for instance in the following cases:

- If we consider the objects of a functional data type to be given only up to extensional equality, we cannot treat the intensional properties of functions in the quotient w.r.t. to behavioral equivalence in a final algebra.
- If we want to specify a choice function on the data type of unordered sets, we can only do this with the help of the construction history of sets, e.g. by taking the first element of the list from which the set was constructed. Cf. [Roşu, 2004, § 3.7] for further discussion of this subject.

Due to theses problems we prefer to represent a co-inductive data type by a hidden sort whose constructor sub-universe is an initial term algebra again, but now of all possible constructors for that sort (whatever that means, cf. below). Following [Roşu, 2004], the destructors of that hidden sort will implicitly define the constructors of additional *experimental sorts* from a set $\mathbb{S}_{\mathcal{E}}$, which come with an application function mapping any experiment and any object of hidden sort to a visible sort. Thus, two hidden objects are behaviorally equal iff the results of all their experiments are equal.

The question, however, is what the term "all possible constructors" means. Practically, we may be content with all constructors that we will need in the future. Theoretically, this demand can be satisfied as follows. We assume that the sets $\mathbb{S}$ of sorts of any possible signature S is a subset of some infinite set, say the set $\mathbf{N}_+$ of the positive natural numbers. Then the triples consisting of

- the symbol $\mathcal{C}$,
- a natural number $m \in \mathbf{N}$, and
- a tuple (or non-empty list) $(s_0, \ldots, s_n)$ with $s_i \in \mathbf{N}_+$ (for $i \in \{0, \ldots, n\}$ and $n \in \mathbf{N}$)

can serve as a never to be exhausted reservoir of constructor symbols, simply by declaring

$$(\mathcal{C}, n, (s_0, \ldots, s_n)) : s_0 \to \cdots \to s_n.$$

Note that this construction does not admit us to have sorts that are visible (constructor generated) and hidden (with behavioral equality) in parallel, such as in [Bidoit & Hennicker, 2006]: While it is consistent to assume all possible constructor symbols to be given from the very beginning, it is not consistent to assume all possible specifications for constructors of hidden sorts to given in advance, because they — and especially their behavior — may depend on visible sorts not yet existing.

As we must guarantee that visible sorts are not constructed from hidden sorts whose behavior they again define, we need some we well-founded quasi-ordering $\lesssim$ on the sorts $\mathbb{S}$; i.e. an $\mathbb{S}$-reflexive and transitive binary relation on $\mathbb{S}$. As usual, its (partial) *ordering* $<$ is given as $\lesssim \setminus \gtrsim$, and its *equivalence* $\sim$ is given as $\lesssim \cap \gtrsim$. A data domain of sort s may depend on on the data domain of sort s' only if $s' \lesssim s$. They may be constructed mutually only if $s' \sim s$. Data domains of a visible sort cannot be constructed mutually with those of hidden or experimental sorts.

2 Basic Notions and Notation

We follow standard mathematical writing style, cf. [Gillman, 1987].

We try to be self-contained in this paper. In case we should omit some required information, we refer the reader to the extensive surveys on term rewriting of [Avenhaus & Madlener, 1990], [Dershowitz & Jouannaud, 1990], [Klop, 1992], and [Plaisted, 1993]. Moreover, the introduction to positive/negative-conditional TRSs of [Wirth & Gramlich, 1994a] may be helpful.

'$\mathbf{N}$' denotes the set of natural numbers and '$<$' the ordering on $\mathbf{N}$. The reflexive, symmetric, transitive, and reflexive & transitive closure of a (binary) relation $\longrightarrow$ will be denoted by $\xrightarrow{=}$, $\longleftrightarrow$, $\xrightarrow{+}$, and $\xrightarrow{*}$, respectively. v and w are called *joinable w.r.t.* $\longrightarrow$ if $v{\downarrow}w$, i.e. if $v\xrightarrow{*}\circ\xleftarrow{*}w$.

The *domain* of $\longrightarrow$ is $\quad \mathrm{dom}(\longrightarrow) \;:=\; \{\, s \mid \exists t.\ s{\longrightarrow}t \,\}$

The *image* of a class A under $\longrightarrow$ is $\quad \langle A\rangle{\longrightarrow} \;:=\; \{\, b \mid \exists a\in A.\ a{\longrightarrow}b \,\}$

The notation for relations is precisely reused for functions, which are considered to be right-unique binary relations. We use '$\uplus$' for the union of disjoint classes.

2.1 Signatures and [Constructor] Terms

Definition 2.1 (Signature)

A *signature* is an 8-tuple $(\mathbb{S}, \mathbb{S}_{\mathcal{NJ}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$ of a set of *sorts* $\mathbb{S}$, a set of *no-junk sorts* $\mathbb{S}_{\mathcal{NJ}} \subseteq \mathbb{S}$, a set of *no-confusion sorts* $\mathbb{S}_{\mathcal{NC}} \subseteq \mathbb{S}$, a set of *experimental sorts* $\mathbb{S}_{\mathcal{E}} \subseteq \mathbb{S}_{\mathcal{NJ}} \cap \mathbb{S}_{\mathcal{NC}}$, a relation $\lesssim$ on $\mathbb{S}$, a set of *constant function symbols* $\mathbb{F}$, a set of *constant predicate symbols* $\mathbb{P}$ with $\mathbb{F}\cap\mathbb{P}=\emptyset$, and a set of *declarations* $\mathbb{D}$, such that:

1. $\lesssim$ is a quasi-ordering on $\mathbb{S}$, with ordering $<$ and equivalence $\sim$, such that:
   (a) $<$ is well-founded.
   (b) If $s \sim s' \in \mathbb{S}_{\mathcal{NJ}}$, then $s \in \mathbb{S}_{\mathcal{NJ}}$.
   (c) If $s \sim s' \in \mathbb{S}_{\mathcal{NC}}$, then $s \in \mathbb{S}_{\mathcal{NC}}$.
2. For each $f \in \mathbb{F}$, there is at least one declaration in $\mathbb{D}$ of the form
   $$f : s_0 \to \cdots\ s_{n-1} \to s_n,\ \varsigma$$
   with $n \in \mathbf{N}$, with $s_i \in \mathbb{S}$ for each $i \in \{0, \ldots, n\}$, and with $\varsigma \in \{\mathcal{C},\mathcal{G}\} \cup \{\mathcal{D}\}{\times}\mathbf{N}$.
   (a) In this case, for each declaration
   $$f : s_0 \to \cdots\ s_{n-1} \to s',\ \varsigma'$$
   in $\mathbb{D}$, we require $(\varsigma', s') = (\varsigma, s_n)$. [1]
   (b) For each declaration of the form
   $$c : s_0 \to \cdots\ s_{n-1} \to s_n,\ \mathcal{C},$$

[1]This means that even in the presence of overloaded constant function symbols, each disambiguated function symbol has a unique declaration and each term has a unique sort.

we require[2] $s_n \in \mathbb{S}_{\mathcal{NJ}} \quad \Rightarrow \quad \forall i \in \{0,\ldots,n\}.\ (s_n \gtrsim s_i)$.
We speak of a declaration *of* $c_{s_0,\ldots,s_{n-1}}$ *as a constructor for the sort* s_n.

(c) For each declaration of the form

$$d : s_0 \to \ \cdots \ s_{n-1} \to s_n,\ \ \mathcal{D}, j,$$

we require[3] $j \in \{0,\ldots,n{-}1\}$ and $\forall i \in \{0,\ldots,n\}.\ (s_j \gtrsim s_i)$.
We speak of a declaration *of* $d_{s_0,\ldots,s_{n-1}}$ *as a destructor of the sort* s_j.

3. For each $P \in \mathbb{P}$, there is at least one declaration in $\mathbb{D}$ of the form

$$P : s_1, \ldots, s_n$$

with $n \in \mathbf{N}$, and with $s_i \in \mathbb{S}$ for each $i \in \{1,\ldots,n\}$.

Definition 2.2 (Variable-System, V**,** X**,** $\mathcal{V}_k(\cdots)$**)**

A *variable-system* for a signature $(\mathbb{S}, \mathbb{S}_{\mathcal{NJ}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$ is a $\{\mathcal{C},\mathcal{G}\}{\times}\mathbb{S}$-sorted family

$$\mathrm{V} \ = \ \left(\mathrm{V}_{\varsigma,s}\right)_{(\varsigma,s)\,\in\,\{\mathcal{C},\mathcal{G}\}\times\mathbb{S}}$$

of infinite sets $\mathrm{V}_{\varsigma,s}$ that are mutually disjoint and disjoint from $\mathbb{F}$.
The variables from $\mathrm{V}_{\mathcal{C},s}$ and $\mathrm{V}_{\mathcal{G},s}$ are called the *constructor variables* and the *general variables of sort* s, respectively.

By abuse of notation we sometimes write V to denote $\bigcup_{(\varsigma,s)\,\in\,\{\mathcal{C},\mathcal{G}\}\times\mathbb{S}} \mathrm{V}_{\varsigma,s}$ and

$$\mathrm{V}_\varsigma$$

to denote $\bigcup_{s\,\in\,\mathbb{S}} \mathrm{V}_{\varsigma,s}$.

And when we speak of a set of variables $\mathrm{X} \subseteq \mathrm{V}$, we mean a $\{\mathcal{C},\mathcal{G}\}{\times}\mathbb{S}$-sorted family $\mathrm{X} \ = \ \left(\mathrm{X}_{\varsigma,s}\right)_{(\varsigma,s)\,\in\,\{\mathcal{C},\mathcal{G}\}\times\mathbb{S}}$ with $\mathrm{X}_{\varsigma,s} \subseteq \mathrm{V}_{\varsigma,s}$ for all $(\varsigma, s) \in \{\mathcal{C},\mathcal{G}\}{\times}\mathbb{S}$.

For $k \in \{\mathcal{C},\mathcal{G}\}{\times}\mathbb{S}$ and for $\mathit{\Gamma}$ being a term, formula, sequent, &c., we use

$$\mathcal{V}_k(\mathit{\Gamma})$$

to denote the set of variables from V_k occurring in $\mathit{\Gamma}$.

---

[2] The idea is that the constructors of no-junk sorts equivalent via $\sim$ construct their domains in parallel on the basis of the domains of those of their arguments whose sort is smaller w.r.t. $<$. Constructors of a junk sort $s \in \mathbb{S} \backslash \mathbb{S}_{\mathcal{NJ}}$, however, may have arguments of a sort s' defined later than s, where even $s{<}s'$ may be the case.

[3] $s_j{>}s_n$ means that s_n serves as a visible sort for experiments on the sort s_j via d. And $s_j{\sim}s_n$ means that experiments on the sort s_n result in experiments on the sort s_j via d. The reason for taking the same hierarchy for hidden and visible sorts is that we do not want to have a visible sort depending on a hidden sort whose visibility it defines. The reason for requiring $s_j{\gtrsim}s_i$ also for $i \prec n$ is the constructor hierarchy of the experimental sorts.

Definition 2.3 ([Constructor] Terms, $\mathrm{Top}(\cdots)$**, Length, Groundness, Linearity)**

Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.
Let $\mathrm{X} \subseteq \mathrm{V}$. Let $s \in \mathbb{S}$. The set

$$\mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{G},s}$$

of *(general) terms of sort s* denotes the disambiguated well-sorted first-order terms of over S and X of sort s. This means that, for all $s' \in \mathbb{S}$, the sets $\mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{G},s'}$ are the smallest sets such that

1. for $x \in \mathrm{X}_{\mathcal{C},s'} \uplus \mathrm{X}_{\mathcal{G},s'}$ we have
$$x \in \mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{G},s'}$$
with *top symbol*
$$\mathrm{Top}(x) = x,$$
and,

2. for each declaration $f : s_0 \to \cdots\ s_{n-1} \to s',\ \varsigma$ in $\mathbb{D}$, and for each $t_0, \ldots, t_{n-1}$ with $t_i \in \mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{G},s_i}$ for each $i \in \{0, \ldots, n-1\}$, we have
$$f_{s_0,\ldots,s_{n-1}}(t_0, \ldots, t_{n-1}) \in \mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{G},s'},$$
with *top symbol*
$$\mathrm{Top}(f_{s_0,\ldots,s_{n-1}}(t_0, \ldots, t_{n-1})) = f_{s_0,\ldots,s_{n-1}}.$$

$\mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C},s}$, however, is the set of *constructor terms of sort s*. This means that, for all $s' \in \mathbb{S}$, the sets $\mathcal{T}(\mathrm{S},\mathrm{V})_{\mathcal{C},s'}$ are the smallest sets such that

1. $$\mathrm{X}_{\mathcal{C},s'} \quad\subseteq\quad \mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C},s'}$$
and,

2. for each declaration $c : s_0 \to \cdots\ s_{n-1} \to s',\ \mathcal{C}$ in $\mathbb{D}$, and for each $t_0, \ldots, t_{n-1}$ with $t_i \in \mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C},s_i}$ for each $i \in \{0, \ldots, n-1\}$, we have
$$c_{s_0,\ldots,s_{n-1}}(t_0, \ldots, t_{n-1}) \in \mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C},s'}.$$

All in all, they form the *term-system*

$$\mathcal{T}(\mathrm{S},\mathrm{X}) \quad=\quad \left(\mathcal{T}(\mathrm{S},\mathrm{X})_{\varsigma,s}\right)_{(\varsigma,s)\,\in\,\{\mathcal{C},\mathcal{G}\}\times\mathbb{S}}.$$

We apply the abuses of notation for variable-systems also to term-systems.

The *length of a term* is $|x| = 1$ for $x \in \mathrm{V}$, and $|f(t_1, \ldots, t_m)| = 1 + |t_1| + \cdots + |t_m|$.

A term is *ground* if no variable occurs in it.

A term is *linear* if each variable occurs at most once in it.

Definition 2.4 (Positions $\mathcal{POS}(\cdots)$)

The set of *positions* of a term t is denoted by $\mathcal{POS}(t)$, and inductively defined as follows. Positions are lists of positive natural numbers.

The *root* or *top position* is the empty list $\emptyset$, i.e.

$$\emptyset \in \mathcal{POS}(t),$$

$$t/\emptyset = t,$$

and

$$t[\,\emptyset \leftarrow s\,] = s.$$

Moreover, for $i \in \{1,\ldots,n\}$,

$$ip \in \mathcal{POS}(f(t_1,\ldots,t_n)) \text{ if } p \in \mathcal{POS}(t_i)$$

$$f(t_1,\ldots,t_n)/ip = t_i/p,$$

and

$$f(t_1,\ldots,t_n)[\,ip \leftarrow s\,] = f(t_1,\ldots,t_{i-1},t_i[\,p \leftarrow s\,],t_{i+1},\ldots,t_n).$$

Definition 2.5 (Constant-Equality)
Two terms $t_0, t_1 \in \mathcal{T}(\mathrm{S},\mathrm{V})$ are *constant-equal* if we have $\mathcal{POS}(t_0) = \mathcal{POS}(t_1)$ and for all $p \in \mathcal{POS}(t_1)$:

1. If $\mathrm{Top}(t_0/p) = f_{s_0,\ldots,s_{n-1}}$ for some constant symbol $f \in \mathbb{F}$ and some sorts $s_i \in \mathbb{S}$ for each $i \in \{0,\ldots,n{-}1\}$; then $\mathrm{Top}(t_0/p) = \mathrm{Top}(t_1/p)$.

2. If $\mathrm{Top}(t_0) \in \mathrm{V}_{\mathcal{G},s} \uplus \mathrm{V}_{\mathcal{C},s}$ for some $s \in \mathbb{S}$, then $\mathrm{Top}(t_1) \in \mathrm{V}_{\mathcal{G},s} \uplus \mathrm{V}_{\mathcal{C},s}$ as well.

Note that Item 2 of Definition 2.5 could be omitted because we have used disambiguated constant function symbols in Item 1.

2.2 Substitutions

Definition 2.6 (Substitution)
Let σ be a *substitution*, i.e. a function from variables to terms. We say that σ is a *substitution on* Y if $\mathrm{dom}(\sigma) \subseteq Y$. Note that for $\sigma := \{x \mapsto x\}$, we have $\mathrm{dom}(\sigma) = \{x\}$ instead of $\mathrm{dom}(\sigma) = \emptyset$. If we declare a substitution as $\sigma : Y \to \mathcal{T}(\mathrm{S},\mathrm{X})$, this means that $\mathrm{dom}(\sigma) = Y$ and that $\mathcal{T}(\mathrm{S},\mathrm{X})$ is the co-domain of σ. With '$\Gamma\sigma$' we denote the result of replacing each occurrence of a variable $x \in \mathrm{dom}(\sigma)$ in Γ with $\sigma(x)$.

Requirement 2.7 ($\mathrm{V}_\mathcal{C}$)
Any substitution σ respects sorts and subsorts:
$\langle \mathrm{V}_{\varsigma,s} \rangle \sigma \subseteq \mathcal{T}(\mathrm{S},\mathrm{V})_{\varsigma,s}$ *for each* $\varsigma \in \{\mathcal{C},\mathcal{G}\}$ *and each* $s \in \mathbb{S}$.

Corollary 2.8 *For* $u \in \mathcal{T}(\mathrm{S},\mathrm{V})_{\mathcal{C},s}$ *and* $\sigma : \mathcal{V}(u) \to \mathcal{T}(\mathrm{S},\mathrm{X})$, *we have* $u\sigma \in \mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C},s}$.

A substitution σ is *a most general unifier of two terms* u_0, u_1 *on* Y if $u_0\sigma = u_1\sigma$ and, for all $\mu : \mathrm{V} \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ with $u_0\mu = u_1\mu$, there is a $\nu : \mathrm{V} \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ such that $(x\sigma)\nu = x\mu$ for all $x \in Y$. Note that a most general unifier always exists for unifiable terms, although each of our types has a single constructor subtype. Indeed, in the subsuming framework of order-sorted signatures, our signatures are "downward unique" and thus admit unitary unification according to Theorem 9.5 of [Waldmann, 1989b]. Our framework may be indeed represented as an instance of the *order-sorted* framework of [Gogolla, 1983; 1984], [Smoka &al., 1989], and [Waldmann, 1989a], roughly as follows (cf. Definition 2.1 for more details):

> No other sub-sort relations are given but
> $$(\mathcal{C}, s) \prec (\mathcal{G}, s)$$
> for each $s \in \mathbb{S}$.
>
> A declaration of a constant symbol f according to
> $$f : s_0 \to \ \cdots \ s_{n-1} \to s_n, \ \varsigma$$
> is to result in the declaration
> $$f : (\mathcal{G}, s_0) \to \ \cdots \ (\mathcal{G}, s_{n-1}) \to (\mathcal{G}, s_n).$$
> Moreover, if $f_{s_0,\ldots,s_{n-1}}$ is a *constructor* symbol (i.e. if $\varsigma = \mathcal{C}$), this is to be expressed in the order-sorted framework with the additional declaration
> $$f : (\mathcal{C}, s_0) \to \ \cdots \ (\mathcal{C}, s_{n-1}) \to (\mathcal{C}, s_n).$$
> A *constructor variable* $x \in \mathrm{V}_{\mathcal{C},s'}$ is declared as $x : (\mathcal{C}, s')$. And a *general variable* $Y \in \mathrm{V}_{\mathcal{G},s}$ is declared as $Y : (\mathcal{G}, s)$.
>
> Let $\mathrm{X} \subseteq \mathrm{V}$. Then $\mathcal{T}(\mathrm{S}, \mathrm{X})_{\mathcal{G},s}$ is the set of terms of the sort $(\mathcal{G}, s)$ in the order-sorted framework, where $\mathcal{T}(\mathrm{S}, \mathrm{X})_{\mathcal{C},s}$ is just the subset of terms $t : (\mathcal{C}, s)$.

Cf. [Wirth, 2008, Note 8] for a general way to model our framework with many-sorted specifications without constructor variables, which uses additional sorts with embedding functions for the constructor sub-sorts, but results in less readable terms.

Replacing all variables from $\mathrm{V}_{\mathcal{C},s} \uplus \mathrm{V}_{\mathcal{G},s}$ with the same chosen variable $x_s \in \mathrm{V}_{\mathcal{C},s}$, we get:

Corollary 2.9 *If* t_0, t_1 *are constant-equal, then there is some* $\mu : \mathcal{V}(t_0, t_0) \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ *such that* $t_0\mu = t_1\mu$.

2.3 Pre-Structures, Valuations, Evaluation

Definition 2.10 (S-**Pre-Structure)**

Let $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$ be a signature.

An S-*pre-structure* $\mathcal{S}$ is a function with the following properties:

1. For each $s \in \mathbb{S}$, $\mathcal{S}$ is defined on $(\mathcal{C}, s)$ and on $(\mathcal{G}, s)$. The values $(\mathcal{C}, s)^{\mathcal{S}}$ and $(\mathcal{G}, s)^{\mathcal{S}}$ are called the *constructor universe* and the *(general) universe of the sort* s, respectively, two sets that satisfy
$$\emptyset \;\neq\; (\mathcal{C}, s)^{\mathcal{S}} \;\subseteq\; (\mathcal{G}, s)^{\mathcal{S}}.$$
2. For each declaration $f : s_0 \to \cdots s_{n-1} \to s_n,\ \varsigma$ in $\mathbb{D}$, $\mathcal{S}$ is defined on $f_{s_0,\ldots,s_{n-1}}$. The value is a function $f^{\mathcal{S}}_{s_0,\ldots,s_{n-1}}$ such that
   - $$f^{\mathcal{S}}_{s_0,\ldots,s_{n-1}} : (\mathcal{G}, s_0)^{\mathcal{S}} \times \cdots \times (\mathcal{G}, s_{n-1})^{\mathcal{S}} \to (\mathcal{G}, s_n)^{\mathcal{S}};$$
   - and, in case of $\varsigma = \mathcal{C}$, also
   $$\langle (\mathcal{C}, s_0)^{\mathcal{S}} \times \cdots \times (\mathcal{C}, s_{n-1})^{\mathcal{S}} \rangle f^{\mathcal{S}}_{s_0,\ldots,s_{n-1}} \;\subseteq\; (\mathcal{C}, s_n)^{\mathcal{S}}.$$
3. For each declaration $P : s_0, \ldots, s_{n-1}$ in $\mathbb{D}$, $\mathcal{S}$ is defined on $P_{s_0,\ldots,s_{n-1}}$. The value is a set $P^{\mathcal{S}}_{s_0,\ldots,s_{n-1}}$ such that
$$P^{\mathcal{S}}_{s_0,\ldots,s_{n-1}} \;\subseteq\; (\mathcal{G}, s_0)^{\mathcal{S}} \times \cdots \times (\mathcal{G}, s_{n-1})^{\mathcal{S}}.$$

Definition 2.11 ($\mathcal{S}$-Valuation)

Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.

Let $\mathcal{S}$ be a S-pre-structure.

An $\mathcal{S}$-*valuation* τ [*on* X] is a (possibly) partial function on V with [$\mathrm{dom}(\tau) = \mathrm{X}$ and] $\tau(x) \in (\varsigma, s)^{\mathcal{S}}$ for each $x \in \mathrm{V}_{\varsigma,s} \cap \mathrm{dom}(\tau)$ and each $(\varsigma, s) \in \{\mathcal{C}, \mathcal{G}\} \times \mathbb{S}$.

We denote with $\tau : \mathrm{X} \to \mathcal{S}$ an $\mathcal{S}$-valuation τ on X, and with $\tau : \mathrm{X} \rightsquigarrow \mathcal{S}$ an $\mathcal{S}$-valuation τ on some Y with $\mathrm{Y} \subseteq \mathrm{X}$.

Definition 2.12 (Evaluation Function “eval”**)**

Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.

Let $\mathcal{S}$ be an S-pre-structure. Let τ be an $\mathcal{S}$-valuation.

The *evaluation function* “eval” is defined as follows:

- For $x \in \mathrm{V}$ we have $\mathrm{eval}(\mathcal{S} \uplus \tau)(x) = \tau(x)$.
- For $f \in \mathbb{F}$ with a declaration of the form $f : s_0 \to \cdots \to s_{n-1} \to s'',\ \varsigma$ in $\mathbb{D}$, and for terms $t'_0, \ldots, t'_{n-1}$ with $t'_i \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{G}, s_i}$ and $s_i \in \mathbb{S}$ for each $i \in \{0, \ldots, n{-}1\}$:
$$\mathrm{eval}(\ \mathcal{S} \uplus \tau\ )(\ f_{s_0,\ldots,s_{n-1}}(t'_0, \ldots, t'_{n-1})\ ) \quad = \quad f^{\mathcal{S}}_{s_0,\ldots,s_{n-1}} \left(\begin{array}{l} \mathrm{eval}(\mathcal{S} \uplus \tau)(t'_0), \\ \ldots, \\ \mathrm{eval}(\mathcal{S} \uplus \tau)(t'_{n-1}) \end{array} \right).$$

- For $P \in \mathbb{P}$ with a declaration of the form $P : s_1, \ldots, s_n$ in $\mathbb{D}$, and for terms $t'_1, \ldots, t'_n$ with $t'_i \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{G}, s_i}$ and $s_i \in \mathbb{S}$ for each $i \in \{1, \ldots, n\}$:

$$\mathrm{eval}\Big(\ \mathcal{S}\uplus\tau\ \Big)\Big(\ P_{s_1,\ldots,s_n}(t'_1, \ldots, t'_n)\ \Big) \quad = \quad \left\{ \begin{array}{ll} \mathsf{TRUE} & \text{if } \left(\begin{array}{l} \mathrm{eval}(\mathcal{S}\uplus\tau)(t'_1), \\ \ldots, \\ \mathrm{eval}(\mathcal{S}\uplus\tau)(t'_n) \end{array} \right) \in P^{\mathcal{S}}_{s_1,\ldots,s_n} \\ \mathsf{FALSE} & \text{otherwise} \end{array} \right\}.$$

- For $s \in \mathbb{S}$ and for $t_0, t_1 \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{G}, s}$:

$$\mathrm{eval}(\mathcal{S}\uplus\tau)(t_0{=}t_1) \quad = \quad \left\{ \begin{array}{ll} \mathsf{TRUE} & \text{if } \mathrm{eval}(\mathcal{S}\uplus\tau)(t_0) = \mathrm{eval}(\mathcal{S}\uplus\tau)(t_1) \\ \mathsf{FALSE} & \text{otherwise} \end{array} \right\}.$$

2.4 Constructing Terms

Definition 2.13 ([Extended] Constructing Terms $\mathcal{CT}$(S,V), $\mathcal{ECT}$(S,V))

Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$. Let $s' \in \mathbb{S}$.

The set $\mathcal{CT}(\mathrm{S,V})_{s'}$ of *constructing terms of the sort* s'
(or else: the set $\mathcal{ECT}(\mathrm{S,V})_{s'}$ of *extended constructing terms of the sort* s')
is given as follows:

$t \in \mathcal{CT}(\mathrm{S,V})_{s'}$ (or else: $t \in \mathcal{ECT}(\mathrm{S,V})_{s'}$) if $t \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{G}, s'}$ and, for each $p \in \mathcal{POS}(t)$, the following holds:

- If $\mathrm{Top}(t/p) = f_{s_0,\ldots,s_{n-1}}$ for some constant symbol $f \in \mathbb{F}$ and some sorts $s_i \in \mathbb{S}$ for each $i \in \{0, \ldots, n{-}1\}$;
  then, for the unique (ς, s'') such that $\mathbb{D}$ contains a declaration of the form
  $$f : s_0 \to \ \cdots \ \to s_{n-1} \to s'',\ \varsigma,$$
  we have $s'' \sim s'$ and $\varsigma = \mathcal{C}$ (or else: $s'' \in \mathbb{S}$ and $\varsigma = \mathcal{C}$).

- If $\mathrm{Top}(t/p) = x$ for some variable $x \in \mathrm{V}$;
  then, for the unique (ς, s'') such that $x \in \mathrm{V}_{\varsigma, s''}$, we have $s'' < s'$ and $\varsigma = \mathcal{C}$ (or else: $s'' \in \mathbb{S}$ and $\varsigma \in \{\mathcal{C}, \mathcal{G}\}$).

2.5 Built-In Terms and Predicates

As the constructor terms of a sort $s \in \mathbb{S}_{\mathcal{NT}} \cap \mathbb{S}_{\mathcal{NC}}$ are to be identical to their constructor sub-universes up to isomorphism, with the exception of their sub-terms of sorts not in $\mathbb{S}_{\mathcal{NT}} \cap \mathbb{S}_{\mathcal{NC}}$, these "built-in terms" already represent their constructor sub-universes one-to-one and may be used to define "built-in" predicates that can be guaranteed to be somehow the same in all possible structures.

Definition 2.14 ([Extended] Built-In Terms $\mathcal{BT}(\mathrm{S},\mathrm{V})$)

Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$. Let $s' \in \mathbb{S}$.

The set $\mathcal{BT}(\mathrm{S},\mathrm{V})_{s'}$ of *build-in terms of the sort s'*
(or else: the set $\mathcal{EBT}(\mathrm{S},\mathrm{V})_{s'}$ of *extended build-in terms of the sort s'*)
is given as follows:

$t \in \mathcal{BT}(\mathrm{S},\mathrm{V})_{s'}$ (or else: $t \in \mathcal{EBT}(\mathrm{S},\mathrm{V})_{s'}$) if $t \in \mathcal{T}(\mathrm{S},\mathrm{V})_{\mathcal{G},s}$ and, for each $p \in \mathcal{POS}(t)$, the following holds:

- If $\mathrm{Top}(t/p) = f_{s_0,\ldots,s_{n-1}}$ for some constant symbol $f \in \mathbb{F}$ and some sorts $s_i \in \mathbb{S}$ for each $i \in \{0,\ldots,n{-}1\}$;
  then, for the unique (ς, s'') such that $\mathbb{D}$ contains a declaration of the form
  $$f : s_0 \to \cdots \to s_{n-1} \to s'', \ \varsigma,$$
  we have $s'' \in \mathbb{S}_{\mathcal{NT}} \cap \mathbb{S}_{\mathcal{NC}}$ (or else: $s'' \in \mathbb{S}_{\mathcal{NC}}$) and $\varsigma = \mathcal{C}$.

- If $\mathrm{Top}(t/p) = x$ for some variable $x \in \mathrm{V}$;
  then, for the unique (ς, s'') such that $x \in \mathrm{V}_{\varsigma,s''}$,
  we have $s'' \notin \mathbb{S}_{\mathcal{NT}} \cap \mathbb{S}_{\mathcal{NC}}$ (or else: $s'' \notin \mathbb{S}_{\mathcal{NC}}$) and $\varsigma = \mathcal{G}$.

Note that we have $\mathcal{CT}(\mathrm{S},\mathrm{V}) \subseteq \mathcal{ECT}(\mathrm{S},\mathrm{V})$, $\mathcal{BT}(\mathrm{S},\mathrm{V}) \subseteq \mathcal{ECT}(\mathrm{S},\mathrm{V})$, and $\mathcal{EBT}(\mathrm{S},\mathrm{V}) \subseteq \mathcal{ECT}(\mathrm{S},\mathrm{V})$, but not other subset relations hold in general for these sets. As the sorts of the variables are different from the sorts of the function symbols for any T with $T \in \{\mathcal{CT}(\mathrm{S},\mathrm{V}), \mathcal{BT}(\mathrm{S},\mathrm{V}), \mathcal{EBT}(\mathrm{S},\mathrm{V})\}$, we get the following corollary, which, however, does not hold for $\mathcal{ECT}(\mathrm{S},\mathrm{V})$ in general.

Corollary 2.15 *Let $T \in \{\mathcal{CT}(\mathrm{S},\mathrm{V}), \mathcal{BT}(\mathrm{S},\mathrm{V}), \mathcal{EBT}(\mathrm{S},\mathrm{V})\}$. We have:*

1. *$T \subseteq \mathcal{ECT}(\mathrm{S},\mathrm{V})$.*
2. *If, for each $i \in \{0,1\}$, we have $\hat{u}_i \in T$ and $\varrho_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S},\mathrm{V})$, and if $\hat{u}_0\varrho_0 = \hat{u}_1\varrho_1$; then $\hat{u}_0, \hat{u}_1$ are constant-equal.*

Note that Item 2 of Corollary 2.15 serves as a strengthened converse of Corollary 2.9.

Definition 2.16 (Built-In Predicates)

Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.

Let $P \in \mathbb{P}$ with declaration $P : s_1, \ldots, s_n$ in $\mathbb{D}$.

Now the disambiguated predicate symbol $P_{s_1,\ldots,s_n}$ is a *built-in predicate for* S if there is associated with it a fixed subset of $\mathcal{BT}(\mathrm{S},\mathrm{V})_{s_1} \times \cdots \times \mathcal{BT}(\mathrm{S},\mathrm{V})_{s_n}$, which is closed under constant-equality. By abuse of notation we denote this set with $P_{s_1,\ldots,s_n}$ as well.

2.6 Structures

Definition 2.17 (S-**structure**)
Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.
An S-structure $\mathcal{S}$ is an S-pre-structure that satisfies following properties:

1. In case of $s \in \mathbb{S}_{\mathcal{NT}}$:
$$(\mathcal{C}, s)^{\mathcal{S}} \quad = \quad \{\ \mathrm{eval}(\mathcal{S} \uplus \tau)(t) \mid t \in \mathcal{CT}(\mathrm{S},\mathrm{V})_s \quad \wedge \quad \tau : \mathcal{V}(t) \to \mathcal{S}\ \}.$$

2. In case of $s \in \mathbb{S}_{\mathcal{NC}}$, $t_i \in \mathcal{CT}(\mathrm{S},\mathrm{V})_s$, $\tau_i : \mathcal{V}(t_i) \to \mathcal{S}$ for $i \in \{0, 1\}$, and
$$\mathrm{eval}(\mathcal{S} \uplus \tau_0)(t_0) = \mathrm{eval}(\mathcal{S} \uplus \tau_1)(t_1);$$
then we t_0, t_1 are constant-equal, and if $\mathrm{Top}(t_0/p) \in \mathrm{V}$, then $\tau_0(t_0/p) = \tau_1(t_1/p)$.

3. Let $P \in \mathbb{P}$ with declaration $P : s_1, \ldots, s_n$ in $\mathbb{D}$. If $P_{s_1,\ldots,s_n}$ is a built-in predicate, then we require for $o_1, \ldots, o_n$ with $o_i \in (\mathcal{G}, s_i)^{\mathcal{S}}$ for each $i \in \{1, \ldots, n\}$:
$$(o_0, \ldots, o_{n-1}) \in P^{\mathcal{S}}_{s_1,\ldots,s_n} \text{ iff}$$
there are an index set $I \subseteq \{1, \ldots, n\}$ and some terms $t_i \in \mathcal{BT}(\mathrm{S},\mathrm{V})_{s_i}$ and some valuations $\tau_i : \mathcal{V}(t_i) \to \mathcal{S}$, such that $\mathrm{eval}(\mathcal{S} \uplus \tau_i)(t_i) = o_i$ for each $i \in I$ and
$$A_1 \times \cdots \times A_n \ \subseteq\ P_{s_1,\ldots,s_n},$$
where $A_i := \left\{ \begin{array}{ll} \{t_i\} & \text{if } i \in I \\ \mathcal{BT}(\mathrm{S},\mathrm{V})_{s_i} & \text{if } i \in \{1, \ldots, n\} \backslash I \end{array} \right\}.$

3 Conditional Term Rewriting Systems

3.1 Motivation

Recursion is a form of programming or definition where a newly defined notion may even occur in its *definientia*. Contrary to *explicit* definitions, where we can always get rid of the new notions by reduction, i.e. by rewriting the *definienda* (*left-hand sides* of the defining equations) to the *definientia* (*right-hand sides*), reduction with *recursive* definitions may run forever.

By formalizing recursive definitions with left-linear term rewriting systems (*TRS*s) we can stay very close to the original intuition.

- From a *theoretical* point of view, recursion can be modeled with unconditional TRSs. Indeed, each Turing machine (including a universal one) can easily be programmed as a monadic, left-linear, unconditional TRS whose critical overlaps are trivial self-overlays; cf. [Huet & Lankford, 1978].
- From a *practical* point of view, however, left-linear positive/negative-conditional TRSs with non-trivial critical overlaps are more convenient and in great demand; cf. [Kaplan, 1988], [Bachmair & Ganzinger, 1991], [Becker, 1993], [Wirth & Lunde, 1994], [Wirth & Gramlich, 1994a], and § 4 of [Wirth, 1997].

The problems in finding a meaningful semantics for recursion modeled in the framework of TRSs may seem just as difficult as for untyped λ-calculus, cf. [Barendregt, 1984]. To state our case clearly, let us name these problems for the instance of first-order arithmetic of natural numbers, assuming the consistency of arithmetic on the meta-level in the sense of [Gentzen, 1938]. The consistency of arithmetic might be destroyed on the object-level by the following two dangers:

Non-Termination: Even if we require that each new term is reducible, some non-terminating recursions (such as $\mathsf{a} = \mathsf{a}+1$) may introduce error objects which cannot be consistently identified with any natural number.

Inconsistent Identification: A function-call may evaluate non-confluently to two different natural numbers, resulting in their equality, such as for $\mathsf{a} = 0$ and $\mathsf{a} = 1$.

On the one hand, programming languages — such as HASKELL (cf. e.g. [Hudlak &al., 1999]) or CURRY (cf. e.g. [Hanus, 2006]) — circumvent these dangers by tailored semantics, such as priority rewriting and "non-deterministic functions". Theoreticians, on the other hand, tend to avoid these dangers by proving some *recursion theorems*, saying that certain restricted forms of recursive equations have unique solutions in terms of total functions for the recursive operators. These unique solutions can then be thought to define these operators explicitly, and dangers cannot occur. For instance, Slind [1996] presents the idea that recursive functions must not be specified without a well-founded ordering, and that the non-decreasing calls in the function definition's right-hand sides are to be immediately evaluated to $\perp$ and not rewritten recursively.

- From a *conceptual* point of view, recursion theorems are counterintuitive because rewriting with the defining equations is not modeled at all. We believe that the recursive process of rewriting *definienda* to *definientia* is the conceptual basis for both recursive specification and computation. Indeed, for studying what could go wrong with recursive definitions, we have to model this *recursive rewriting* explicitly.

- From a *practical* point of view, recursion theorems such as the one of Slind [1996] are infeasible because many recursive programs (such as interpreters) — by their very nature — do not terminate on all input and have undecidable termination problems; and even if they do terminate, neither the termination nor the well-founded ordering to show it may be known at definition time.

Note that the recursion theorem of Slind [1996] is not a poor instance of a recursion theorem, but — to the best of our knowledge — one of the strongest and practically most adequate approaches to admit recursive definitions via recursion theorems. It is implemented e.g. in ISABELLE/HOL, cf. e.g. [Nipkow &al., 2000; 2002].

In this paper we are interested in a conceptually and practically adequate treatment of recursion via rewriting.

The state of the art of controlling the danger of inconsistent identification are the confluence criteria of [Wirth, 2008]. In this paper we will simply assume confluence to be given.

Our answer to the danger of non-termination is simple: Don't worry, be happy! Indeed, an arbitrary number of error objects is beneficial to the semantics of so-called *data models*. Kühler & Wirth [1996] describe this semantics under a practical aspect. Under a more theoretical and general aspect, Wirth & Gramlich [1994b] study this and other possible semantics for partially specified functions. These studies are extended in § 10 of [Wirth, 1997]. *Validity in all data models*, however, still appears as the most natural semantical answer to the danger of non-termination. But we will add a lot to that answer in this paper.

3.2 Basic Notions and Notation

A *(conditional) equation* is a triple $l{=}r{\longleftarrow}C$ consisting of two terms of the same sort, namely the *left-hand side* l and the *right-hand side* r, and the *condition* C. It is to be interpreted as: "For all variables from $\mathcal{V}(l{=}r{\longleftarrow}C)$, the left-hand side l is equal to the right-hand side r if the condition C holds." Cf. [Kühler & Wirth, 1996], [Wirth, 1997], and [Wirth & Gramlich, 1994b] for more on semantics. If the condition is empty, i.e. if $C=\emptyset$, the equation is *unconditional* and we simply write $l{=}r$ instead of $l{=}r{\longleftarrow}C$. An *(unconditional) TRS* is a set of unconditional equations. A *conditional TRS* (*CTRS*) is a set of (conditional) equations.

A CTRS is *ground* if all its terms are ground. A CTRS is *left-linear* if all its left-hand sides are linear terms.

We are going to define a term rewrite or reduction relation for a CTRS R and a set of variables X $\subseteq$ V. As we will not instantiate the variables from X in the reduction relation (as would be required for considering stability under substitution), the elements of X are to be considered as unknowns, parameters, or free generators. For instance, in QuodLibet, the set X is set to $\mathrm{V}\setminus\mathrm{V}_{\mathcal{C}}$ for semantical reasons described by [Kühler & Wirth, 1996] ($\overset{*}{\longleftrightarrow}_{\mathrm{R,X}}$ becomes equality in all data models), and $\mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C}}$ becomes the set of constructor ground terms.

Definition 3.1 (Reduction Relation, $\longrightarrow_{\mathrm{R,X},i,p}$)
Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.
Let R be a CTRS. Let $\mathrm{X} \subseteq \mathrm{V}$.
Let us define our reduction relation $\longrightarrow_{\mathrm{R,X}}$ top-down, inductively over the *condition-recursion depth* i, and modulo the property of fulfilledness of Definition 3.10.
$\longrightarrow_{\mathrm{R,X}} := \bigcup_{i\in\mathbf{N}} \longrightarrow_{\mathrm{R,X},i}$.
For $s,t \in \mathcal{T}(\mathrm{S},\mathrm{X})$ and $i \in \mathbf{N}$, we define $s \longrightarrow_{\mathrm{R,X},i} t$ if $s \longrightarrow_{\mathrm{R,X},i,p} t$ for some $p \in \mathcal{POS}(s)$.
$\longrightarrow_{\mathrm{R,X},0,p} := \emptyset$.
$s \longrightarrow_{\mathrm{R,X},i+1,p} t$ if there are $(l{=}r{\longleftarrow}C) \in \mathrm{R}$ and a substitution $\mu : \mathcal{V}(l{=}r{\longleftarrow}C) \to \mathcal{T}(\mathrm{S},\mathrm{X})$, such that $s/p = l\mu$ and $t = s[\,p \leftarrow r\mu\,]$ and $C\mu$ is *fulfilled* w.r.t. $\longrightarrow_{\mathrm{R,X},i}$ and S.

The notion of fulfilledness may remain abstract up to § 3.3. The only thing we need right now is that the operator that maps a relation $\longrightarrow$ to the conditions fulfilled w.r.t. $\longrightarrow$ and S is monotonic, i.e. that a bigger $\longrightarrow$ results in more conditions being fulfilled w.r.t. $\longrightarrow$ and S. Then a trivial induction shows:

Corollary 3.2 *For $i,j \in \mathbf{N}$ with $i \leq j$ we have* $\longrightarrow_{\mathrm{R,X},i} \subseteq \longrightarrow_{\mathrm{R,X},j} \subseteq \longrightarrow_{\mathrm{R,X}}$.

Corollary 3.3 (Destructing Derivations)
If $s \longrightarrow_{\mathrm{R,X},n,pq} t$, then $s/p \longrightarrow_{\mathrm{R,X},n,q} t/p$ and $t = s[\,p \leftarrow t/p\,]$.

Corollary 3.4 (Constructing Derivations)

1. *If $t/p \longrightarrow_{\mathrm{R,X},n,q} u$, then we have $t \longrightarrow_{\mathrm{R,X},n,pq} t[\,p \leftarrow u\,]$.*
2. *Let $\Pi \subseteq \mathcal{POS}(t)$ be a parallel set of positions.*
   If $t/p \overset{*}{\longrightarrow}_{\mathrm{R,X},n} u_p$ for each $p \in \Pi$, then we have $t \overset{*}{\longrightarrow}_{\mathrm{R,X},n} t[\,p \leftarrow u_p \mid p \in \Pi\,]$.

A variable in $\mathcal{V}(r,C)\setminus\mathcal{V}(l)$ is called an *extra* variable of the equation $l{=}r{\longleftarrow}C$. According to the occurrences of extra variables in their equations, the following taxonomy of CTRSs (restricting *all* their equations) is not mnemonic but folklore, cf. e.g. [Hanus, 1994; Suzuki &al., 1995]:

1-CTRS No extra variables: $\mathcal{V}(r,C) \subseteq \mathcal{V}(l)$.

2-CTRS Extra variables only in the conditions: $\mathcal{V}(r) \subseteq \mathcal{V}(l)$.

3-CTRS Extra variables must occur in the condition of their equation: $\mathcal{V}(r) \subseteq \mathcal{V}(l,C)$.

4-CTRS No restriction on extra variables.

We will use n-CTRS both as a noun and as an adjective.

Definition 3.5 (Normality; [Bergstra & Klop, 1986])
A CTRS R is *normal* if the right-hand sides u_2 of the equations $(u_1{=}u_2)$ listed in the conditions C of the conditional equations $(l{=}r{\longleftarrow}C) \in \mathrm{R}$ are ground terms which are irreducible w.r.t. R. (For decidability reasons, this irreducibility requirement is typically strengthened to irreducibility w.r.t. the associated unconditional TRS, which has all conditions removed, assuming them to be always fulfilled.)

While the requirement of normality (cf. Definition 3.5) is a strong restriction on the form of CTRSs (cf. the discussions in §§ 6 and 7.3 of [Wirth, 2008]), its weak form of *quasi-normality* only disallows that *both* the left-hand side and the right-hand side of an equation in a condition are undefined objects in the sense of Definition 3.10. Quasi-normality can easily be achieved for constructor-based specifications in practice. Indeed, all confluent example specifications in this paper are quasi-normal.

Definition 3.6 ([Weak-] Quasi-Normality)
Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.
Let R be a CTRS and $\mathrm{X} \subseteq \mathrm{V}$.
A condition C is [*weak-*] *quasi-normal w.r.t.* R, X if for all $(u_0{=}u_1)$ listed in C:

- $[\mathcal{V}(u_0, u_1) \subseteq \mathrm{V}_{\mathcal{C}}$ or$]$
- for all $\mu : \mathcal{V}(C) \to \mathcal{T}(\mathrm{S}, \mathrm{X})$ for which $C\mu$ is fulfilled w.r.t. $\longrightarrow_{\mathrm{R,X}}$ and S, there is an $i \in \{0, 1\}$ such that (Def $u_i\mu$) occurs in $C\mu$ or that $u_i\mu \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{C}}$.

R, X is [*weak-*] *quasi-normal* if
C is [weak-] quasi-normal w.r.t. R, X for all $(l{=}r{\longleftarrow}C) \in \mathrm{R}$.

3.3 Constructors and Fulfilledness

Let us describe the special settings of this paper which are presupposed in the following; namely the irreducibility of constructor terms and the possible conditions of conditional equations and their fulfilledness.

Operationally, the set $\mathcal{T}(\mathrm{S}, \mathrm{X})_{\mathcal{C}}$ of *constructor terms* guarantees irreducibility also in future consistent extensions of the specification. Semantically, $\mathcal{T}(\mathrm{S}, \mathrm{X})_{\mathcal{C}}$ is intended to denote the *defined terms*, *data objects*, or *really existing objects*. For the involved motivations behind this cf. [Hobbs, 1996], [Kühler & Wirth, 1996], [Padawitz, 1988, § 6], and [Padawitz, 2000].

Requirement 3.7 (Requirements 20 and 43 ($\Sigma_{\mathcal{C}}$ and $\Sigma_{\mathcal{NT}}$) of [Wirth, 2008])
For any signature $\mathrm{S} = (\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$, *and for any CTRS* R *and each* $(l{=}r{\longleftarrow}C) \in \mathrm{R}$ *we require* $\mathrm{Top}(l) = f_{s_0,\ldots,s_{n-1}}$, *for some* $f \in \mathbb{F}$ *with declaration* $f : s_0 \to \cdots \to s_{n-1} \to s''$, ς *in* $\mathbb{D}$ *and with* $\varsigma \neq \mathcal{C}$.

By Requirement 3.7, no left-hand side can match a term from $\mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{C}}$, nor from $\mathcal{ECT}(\mathrm{S},\mathrm{V})$. Thus we have:

Corollary 3.8 $\mathrm{dom}(\longrightarrow_{\mathrm{R,X}}) \subseteq \mathcal{T}(\mathrm{S}, \mathrm{X}) \setminus \mathcal{T}(\mathrm{S}, \mathrm{X})_{\mathcal{C}}$.

Corollary 3.9 *Let $\hat{u} \in \mathcal{ECT}(\mathrm{S,V})$, and $\varrho : \mathcal{V}(\hat{u}) \to \mathcal{T}(\mathrm{S,V})$. If $\hat{u}\varrho \xrightarrow{*}_{\mathrm{R,X}} v$, then there is a $\varrho' : \mathcal{V}(\hat{u}) \to \mathcal{T}(\mathrm{S,V})$ with $v = \hat{u}\varrho'$ and $\forall x \in \mathcal{V}(\hat{u}).\ (x\varrho \xrightarrow{*}_{\mathrm{R,X}} x\varrho')$.*

We assume that our conditions are conjunctive lists of literals of the following five forms: $(u_1{=}u_2)$, $(u_1{\neq}u_2)$, $(\mathrm{Def}\ u)$, $P(u_1,\ldots,u_n)$, $\neg P(u_1,\ldots,u_n)$, where P is an n-ary built-in predicate on $\mathcal{BT}(\mathrm{S,V})$, cf. § 2.5. As these predicates are not specified by the CTRSs but must be given by additional means, we have one more reason to call them *built-in predicates*.

As already explained in § 2.5, by abuse of notation, each P has a syntactical and a semantical aspect, which can easily be distinguished in our context even without a different notation: the symbol $P : s_1,\ldots,s_n$ in our formal language (forming the literal $P(u_1,\ldots,u_n)$ for any terms $u_i \in \mathcal{T}(\mathrm{S,X})_{s_i}$); and the relation $P \subseteq \mathcal{BT}(\mathrm{S,V})_{s_1} \times \cdots \times \mathcal{BT}(\mathrm{S,V})_{s_n}$ in the $\mathcal{BT}(\mathrm{S,V})$-reduct of the (sorted) free term algebra over X.

We may have arbitrarily many such predicates P on $\mathcal{BT}(\mathrm{S,V})$ of arbitrary arity, cf. [Wirth, 2008, Note 2].

Definition 3.10 (Fulfilledness)

1. A condition *is fulfilled w.r.t.* $\longrightarrow$ *and* S if all its literals are fulfilled w.r.t. $\longrightarrow$ and S.

2. $(u_1{=}u_2)$ *is fulfilled w.r.t.* $\longrightarrow$ *and* S if $u_1{\downarrow}u_2$.

3. $(u_1{\neq}u_2)$ *is fulfilled w.r.t.* $\longrightarrow$ *and* S if for each $i \in \{0,1\}$, there are $\hat{u}_i \in \mathcal{EBT}(\mathrm{S,V})$ and $\varrho_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S,V})$, such that $u_i \xrightarrow{*} \hat{u}_i\varrho_i$, and $\hat{u}_0, \hat{u}_1$ are not constant-equal.[4]

4. $(\mathrm{Def}\ u)$ *is fulfilled w.r.t.* $\longrightarrow$ *and* S if there is a $\hat{u} \in \mathcal{T}(\mathrm{S,X})_{\mathcal{C}}$ such that $u \xrightarrow{*} \hat{u}$. [5]

5. $P(u_1,\ldots,u_n)$ *is fulfilled w.r.t.* $\longrightarrow$ *and* S if there is an index set $I \subseteq \{1,\ldots,n\}$ such that, setting $J := \{1,\ldots,n\}\backslash I$, for each $i \in I$, there are $\hat{u}_i \in \mathcal{BT}(\mathrm{S,V})$ and $\varrho_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S,V})$ with $u_i \xrightarrow{*} \hat{u}_i\varrho_i$ such that, for all $(\hat{u}_j)_{j\in J}$ with $\forall j \in J.\ (\hat{u}_j \in \mathcal{BT}(\mathrm{S,V}))$, it is the case that $P(\hat{u}_1,\ldots,\hat{u}_n)$ is effectively known to hold.

6. $\neg P(u_1,\ldots,u_n)$ *is fulfilled w.r.t.* $\longrightarrow$ *and* S if there is an index set $I \subseteq \{1,\ldots,n\}$ such that, setting $J := \{1,\ldots,n\}\backslash I$, for each $i \in I$, there are $\hat{u}_i \in \mathcal{BT}(\mathrm{S,V})$ and $\varrho_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S,V})$ with $u_i \xrightarrow{*} \hat{u}_i\varrho_i$, such that, for all $(\hat{u}_j)_{j\in J}$ with $\forall j \in J.\ (\hat{u}_j \in \mathcal{BT}(\mathrm{S,V}))$, it is the case that $P(\hat{u}_1,\ldots,\hat{u}_n)$ is effectively known *not* to hold.

---

[4] Note that, contrary to Item 3 of Definition 22 of [Wirth, 2008], $\neq$ can be seen as the built-in [constant-] inequality predicate on $\mathcal{BT}(\mathrm{S,V})$. Indeed, $\neq$ may exploit inequality of the matching terms from $\mathcal{EBT}(\mathrm{S,V})$, even if matching terms from $\mathcal{BT}(\mathrm{S,V})$ are equal.

[5] Note that, contrary to Item 4 of Definition 22 of [Wirth, 2008], Def ("*definedness*") cannot be seen as the built-in universal predicate on $\mathcal{BT}(\mathrm{S,V})$. Again, however, but now for a different reason, namely because its negation is not consistent w.r.t. consistent extension, $\neg(\mathrm{Def}\ u)$ is not admitted as a condition literal.

The reason why we admit I to be a proper subset of $\{1,\ldots,n\}$ is that, e.g. for

$$P(u_1,u_2,u_3,u_4)$$

being a lexicographic ordering such as

$$(u_1,u_2)<(u_3,u_4),$$

it should be sufficient that u_1 reduces to a shorter term in $\mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C}}$ than u_3, even if u_2 does not reduce to any term of $\mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C}}$ at all. Choose $I:=\{1,3\}$. Then

$$(\mathsf{0},\mathsf{0}-\mathsf{s}(\mathsf{0}))<(\mathsf{s}(\mathsf{0}),\mathsf{0}-\mathsf{s}(\mathsf{s}(\mathsf{0})))$$

is fulfilled, disregarding the undefined terms in the second elements of the pairs, simply because we have

$$(\mathsf{0},\hat{u}_2)<(\mathsf{s}(\mathsf{0}),\hat{u}_4)$$

for all $(\hat{u}_j)_{j\in\{2,4\}}$ with $\forall j\in\{2,4\}.\ (\hat{u}_j\in\mathcal{T}(\mathrm{S},\mathrm{X})_{\mathcal{C}})$.

Let us assume that $\longrightarrow$ is confluent. Then $u_1{\downarrow}u_2$ iff $u_1\overset{*}{\longleftrightarrow}u_2$. Thus, equality literals are tested for equality in the quotient modulo $\overset{*}{\longleftrightarrow}$. Moreover, $P(u_1,\ldots,u_n)$ and $\neg P(u_1,\ldots,u_n)$ cannot both be fulfilled w.r.t. a confluent $\longrightarrow$ and S, because an argument with an index in the intersection of the two index sets would have to have two different normal forms; cf. the proof of Lemma 3.15 for more details. But fulfilledness of $P(u_1,\ldots,u_n)$ and $\neg P(u_1,\ldots,u_n)$ does not have to satisfy *tertium non datur*, cf. [Wirth, 2008, Note 3].

As the reference to the reduction relation $\longrightarrow$ will vary a lot, whereas S will be more constant or clear from the context, in what follows we will often speak of "fulfilledness w.r.t. $\longrightarrow$" where we would actually have to speak of "fulfilledness w.r.t. $\longrightarrow$ and S".

Finally, note that fulfilledness is monotonic, indeed, as presupposed for the validity of Corollary 3.2.

3.4 Compatibility with [Wirth, 2008]

Although, as already said, we will simply assume confluence in this paper, our treatment here would be meaningless in practice unless we show that our new notions of fulfilledness here is compatible with the results of [Wirth, 2008]. Contrary to the notion of fulfilledness of [Wirth, 2008], our notion of fulfilledness here also applies to sorts s with $s\notin\mathbb{S}_{\mathcal{NT}}\cap\mathbb{S}_{\mathcal{NC}}$, and is identical to the one of [Wirth, 2008] only for the case of $\mathbb{S}=\mathbb{S}_{\mathcal{NT}}\cap\mathbb{S}_{\mathcal{NC}}$. Luckily, the whole treatment of fulfilledness in [Wirth, 2008] (with the exception of equality literals) is confined to two lemmas, which we have to re-proof now.

As a corollary of Corollary 3.9 we get:

Corollary 3.11 *Let* $n_0,n_1\in\mathbf{N}$. *Assume that* R, X *is shallow confluent up to* $n_0{+}n_1$. *Let* $u,v\in\mathcal{T}(\mathrm{S},\mathrm{V})$, $\hat{u}\in\mathcal{ECT}(\mathrm{S},\mathrm{V})$, *and* $\varrho:\mathcal{V}(\hat{u})\to\mathcal{T}(\mathrm{S},\mathrm{V})$. *If* $v\overset{*}{\longleftarrow}_{\mathrm{R},\mathrm{X},n_0}u\overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1}\hat{u}\varrho$, *then there is a* $\varrho':\mathcal{V}(\hat{u})\to\mathcal{T}(\mathrm{S},\mathrm{V})$ *with* $v\overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1}\hat{u}\varrho'$.

As a corollary of Corollaries 3.9 and 2.15 we get:

Corollary 3.12 *Let* $T\in\{\mathcal{BT}(\mathrm{S},\mathrm{V}),\ \mathcal{EBT}(\mathrm{S},\mathrm{V})\}$. *For each* $i\in\{0,1\}$, *let* $\hat{u}_i\in T$ *and* $\varrho_i:\mathcal{V}(\hat{u}_i)\to\mathcal{T}(\mathrm{S},\mathrm{V})$. *If* $\hat{u}_0\varrho_0{\downarrow}_{\mathrm{R},\mathrm{X}}\hat{u}_1\varrho_1$, *then* $\hat{u}_0,\hat{u}_1$ *are constant-equal.*

Lemma 3.13 (Variable-Overlap Lemma [Wirth, 2008, Lemma 24])
Let $n_0, n_1 \in \mathbf{N}$. *Let* $(l{=}r{\longleftarrow}C) \in \mathrm{R}$. *Let* $\mu, \nu : \mathcal{V}(l{=}r{\longleftarrow}C) \to \mathcal{T}(\mathrm{S}, \mathrm{X})$.
If R, X *is shallow confluent up to* $n_0{+}n_1$, C *is weak-quasi-normal w.r.t.* R, X, $C\mu$ *is fulfilled w.r.t.* $\longrightarrow_{\mathrm{R},\mathrm{X},n_1}$ *and* S, *and* $\forall x \in \mathcal{V}(C).\ (x\mu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_0} x\nu)$;
then $l\nu \longrightarrow_{\mathrm{R},\mathrm{X},n_1+1} r\nu$ *and* $C\nu$ *is fulfilled w.r.t.* $\longrightarrow_{\mathrm{R},\mathrm{X},n_1}$ *and* S.

Proof of Lemma 3.13 It suffices to show, for each literal L in C, that $L\nu$ is fulfilled w.r.t. $\longrightarrow_{\mathrm{R},\mathrm{X},n_1}$ under the assumption that $C\mu$ is fulfilled w.r.t. $\longrightarrow_{\mathrm{R},\mathrm{X},n_1}$; cf. Definition 3.10.

$\underline{L = (u_0{=}u_1)}$: By $\forall x \in \mathcal{V}(C).\ x\mu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_0} x\nu$ and Corollary 3.4, we get

$$u_0\nu \overset{*}{\longleftarrow}_{\mathrm{R},\mathrm{X},n_0} u_0\mu \downarrow_{\mathrm{R},\mathrm{X},n_1} u_1\mu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_0} u_1\nu.$$

As C is assumed to be weak-quasi-normal w.r.t. R, X, we have to distinguish the following cases: Firstly, in the case of $\mathcal{V}(u_0, u_1) \subseteq \mathrm{V}_{\mathcal{C}}$, by Corollary 2.8 we have for all $x \in \mathcal{V}(u_0, u_1)$: $x\mu \in \mathcal{T}_{\mathcal{C}}(\mathrm{X})$. Then, by Corollaries 3.2 and 3.8, we have $u_0\nu = u_0\mu \downarrow_{\mathrm{R},\mathrm{X},n_1} u_1\mu = u_1\nu$.

Secondly, if there is is an $i \in \{0,1\}$ such that $(\mathrm{Def}\ u_i\mu)$ occurs in $C\mu$ or that $u_i\mu \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{C}}$, then there is a $\hat{v} \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{C}}$ with $u_i\mu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1} \hat{v}$. By Corollary 3.8 and [Wirth, 2008, Lemma 15(4)], we get the desired $u_0\nu \downarrow_{\mathrm{R},\mathrm{X},n_1} u_1\nu$ again.

$\underline{L = (u_0{\neq}u_1)}$: For $i \in \{0,1\}$, there are $\hat{u}_i \in \mathcal{EBT}(\mathrm{S,V})$ and $\varrho_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ with $u_i\nu \overset{*}{\longleftarrow}_{\mathrm{R},\mathrm{X},n_0} u_i\mu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1} \hat{u}_i\varrho_i$ and $\hat{u}_0 \neq \hat{u}_1$. By Corollary 3.11, there are $\varrho'_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ with $u_i\nu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1} \hat{u}_i\varrho'_i$.

$\underline{L = (\mathrm{Def}\ u)}$: Then there is a $\hat{v} \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{C}}$ with $u\nu \overset{*}{\longleftarrow}_{\mathrm{R},\mathrm{X},n_0} u\mu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1} \hat{v}$. By Corollary 3.8 and [Wirth, 2008, Lemma 15(3)], we get $u\nu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1} \hat{v}$.

$\underline{L = [\neg]P(u_1, \ldots, u_k)}$: For $i \in I$, there are $\hat{u}_i \in \mathcal{BT}(\mathrm{S,V})$ and $\varrho_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ with $u_i\nu \overset{*}{\longleftarrow}_{\mathrm{R},\mathrm{X},n_0} u_i\mu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1} \hat{u}_i\varrho_i$ and $P(\hat{u}_1, \ldots, \hat{u}_k)$ [not] holding for all $(\hat{u}_j)_{j \in J}$ with $\forall j \in J.\ (\hat{u}_j \in \mathcal{BT}(\mathrm{S,V}))$. By Corollary 3.11, there are $\varrho'_i : \mathcal{V}(\hat{u}_i) \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ with $u_i\nu \overset{*}{\longrightarrow}_{\mathrm{R},\mathrm{X},n_1} \hat{u}_i\varrho'_i$ for all $i \in I$. **Q.e.d. (Lemma 3.13)**

Definition 3.14 (Complementarity of Condition Lists)
Two condition lists D_0 and D_1 are *complementary* if for a permutation k of $\{0,1\}$, for each $i \in \{0,1\}$, D_{k_i} lists a literal L_i, such that at least one of the following cases holds:

1. L_0 has the form $P(u_1, \ldots, u_m)$ and L_1 has the form $\neg P(u_1, \ldots, u_m)$ for some built-in predicate P on $\mathcal{BT}(\mathrm{S,V})$.

2. L_0 has the form $(u_1{=}u_2)$ and L_1 has the form $(u_1{\neq}u_2)$; and there is an $l \in \{1,2\}$ such that $u_l \in \mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{C}}$ or D_{k_0} lists $(\mathrm{Def}\ u_l)$.

3. Let $T \in \{\mathcal{T}(\mathrm{S}, \mathrm{V})_{\mathcal{C}},\ \mathcal{BT}(\mathrm{S,V}),\ \mathcal{EBT}(\mathrm{S,V})\}$ be fixed.
   Now, for each $i \in \{0,1\}$, L_i has the form $(p{=}c_i)$ with $c_i \in T$,
   such that $c_0\varphi \neq c_1\varphi$ for all $\mu : \mathcal{V}(c_0, c_1) \to \mathcal{T}(\mathrm{S}, \mathrm{X})$.

4. L_i has the form $P_i(y_{i,1}, \ldots, y_{i,m'_i})\sigma$ for distinct variables $y_{i,1}, \ldots, y_{i,m'_i} \in \mathrm{V}$ for each $i \in \{0,1\}$ and some $\sigma : \mathrm{V} \to \mathcal{T}(\mathrm{S}, \mathrm{V})$ such that

$$\neg\exists y_{0,1}, \ldots, y_{0,m'_0}, y_{1,1}, \ldots, y_{1,m'_1} \in \mathcal{BT}(\mathrm{S,V}).\ \left(\begin{array}{cc} & P_0(y_{0,1}, \ldots, y_{0,m'_0}) \\ \wedge & P_1(y_{1,1}, \ldots, y_{1,m'_1}) \end{array}\right)$$

is an element of a fixed decidable set of theorems of the built-in predicates P_0 and P_1 on $\mathcal{BT}$(S,V).

5. L_0 has the form $(y_{0,1}{=}y_{0,2})\sigma$ and L_1 has the form $P(y_{1,1},\ldots,y_{1,m'_1})\sigma$ for variables $y_{i,j}\in\mathrm{V}$ and $\sigma:\mathrm{V}\to\mathcal{T}(\mathrm{S},\mathrm{V})$ such that $y_{1,1},\ldots,y_{1,m'_1}$ are distinct and
$$\neg\exists y_{0,1},y_{0,2},y_{1,1},\ldots,y_{1,m'_1}\in\mathcal{BT}(\mathrm{S},\mathrm{V}).\left(\begin{array}{ll} & (y_{0,1}{=}y_{0,2})\\ \wedge & P(y_{1,1},\ldots,y_{1,m'_1})\end{array}\right)$$
is an element of a fixed decidable set of theorems of the built-in predicate P on $\mathcal{BT}$(S,V); and there is an $l\in\{1,2\}$ such that
$$y_{0,l}\sigma\in\mathcal{T}(\mathrm{S},\mathrm{V})_{\mathcal{C}}\ \text{ or }\ D_{k_0}\text{ lists (Def }y_{0,l}\sigma).$$

Lemma 3.15 ([Wirth, 2008, Lemma 62])
Each binding-complementary critical peak is shallow trivial.

Proof of Lemma 3.15

We omit the subscript R, X in this proof. Assume the situation and notation of Definition 3.14 and [Wirth, 2008, Definition 58]. To show shallow triviality, we assume that the condition of [Wirth, 2008, Definition 30] is satisfied for some $\varphi:\mathrm{V}\to\mathcal{T}(\mathrm{S},\mathrm{X})$ and some $n_0,n_1\in\mathbf{N}$, and will derive a contradiction. Under this assumption, $(C\xi_i)\varphi,(D'_i\xi_i)\varphi$ is fulfilled w.r.t. $\longrightarrow_{n_i}$, and R, X is shallow confluent up to $n_0{+}n_1$. Define $\psi_i:\mathrm{V}\to\mathcal{T}(\mathrm{S},\mathrm{X})$ by $y\psi_i:=y\xi_i\varphi$ for $y\in\{x_1,\ldots,x_m\}\cup Z$, and $y\psi_i:=y\varphi$ for $y\in\mathrm{V}\setminus(\{x_1,\ldots,x_m\}\cup Z)$. Then $C\psi_i,D'_i\psi_i$ is fulfilled w.r.t. $\longrightarrow_{\mathrm{R},\mathrm{X},n_i}$, and $y\psi_0=y\psi_1$ for $y\in\mathrm{V}\setminus(\{x_1,\ldots,x_m\}\cup Z)$. As C contains binding equations for the variables $x_1,\ldots,x_m$ whose left-hand sides do not contain variables from the set Z, by [Wirth, 2008, Lemma 47] we get $x_j\psi_0=x_j\psi_1$ for all $j\in\{1,\ldots,m\}$. Thus, due to $Z\cap\mathcal{V}(D'_i)=\emptyset$, $D'_i\psi_1$ is fulfilled w.r.t. $\longrightarrow_{\mathrm{R},\mathrm{X},n_i}$. By the assumed complementarity there must be complementary literals in D'_0 and D'_1. Thus, assume k to be a permutation of $\{0,1\}$ as described in Definition 3.14. We distinguish the cases of Definition 3.14:

<u>Case 1:</u> When we consider $\neg P$ as the name of the built-in predicate on $\mathcal{BT}$(S,V) complementary to P, by Definition 3.10 this is a subcase of Case 4 for $y_{0,j}=y_{1,j}$ and $y_{0,j}\sigma=u_j$.

<u>Case 2:</u> In this case we have $(u_1{=}u_2)$ in D'_{k_0} and $(u_1{\neq}u_2)$ in D'_{k_1} and there is an $l\in\{1,2\}$ such that (Def u_l) is in D'_{k_0} unless $u_l\in\mathcal{T}(\mathrm{S},\mathrm{V})_{\mathcal{C}}$. By Corollaries 2.8 and 3.8, we get $y_{0,l}\psi_1\overset{*}{\longrightarrow}_{n_{k_0}}\hat{u}\notin\mathrm{dom}(\longrightarrow)$ for some $\hat{u}$.
Moreover, for each $i\in\{0,1\}$, there are $\hat{v}_i\in\mathcal{EBT}(\mathrm{S},\mathrm{V})$ and $\varrho_i:\mathcal{V}(\hat{v}_i)\to\mathcal{T}(\mathrm{S},\mathrm{V})$, such that $\hat{v}_0\neq\hat{v}_1$ are not constant-equal and
$$\hat{v}_0\varrho_0\overset{*}{\longleftarrow}_{n_{k_1}}u_1\psi_1\overset{*}{\longrightarrow}_{n_{k_0}}\circ\overset{*}{\longleftarrow}_{n_{k_0}}u_2\psi_1\overset{*}{\longrightarrow}_{n_{k_1}}\hat{v}_1\varrho_1.$$
By [Wirth, 2008, Lemma 15(4)] and Corollaries 3.2 and 3.8, we get $\hat{v}_0\varrho_0\downarrow_{n_{k_0}}\hat{v}_1\varrho_1$. By Corollaries 3.2 and 3.12 we get the contradictory fact that $\hat{v}_0,\hat{v}_1$ are constant-equal.

<u>Case 3:</u> In this case we have $(p{=}c_i)$ in D'_{k_i}. Thus, for some w_0,w_1, we have
$$c_0\psi_1\overset{*}{\longrightarrow}_{n_{k_0}}w_0\overset{*}{\longleftarrow}_{n_{k_0}}p\psi_1\overset{*}{\longrightarrow}_{n_{k_1}}w_1\overset{*}{\longleftarrow}_{n_{k_1}}c_1\psi_1.$$
By shallow confluence, there is some $\hat{w}$ with
$$c_0\psi_1\overset{*}{\longrightarrow}_{n_{k_0}}w_0\overset{*}{\longrightarrow}_{n_{k_1}}\hat{w}\overset{*}{\longleftarrow}_{n_{k_0}}w_1\overset{*}{\longleftarrow}_{n_{k_1}}c_1\psi_1.$$

Thus, by Corollary 3.2 $c_0\psi_1{\downarrow}c_1\psi_1$. In case of $T=\mathcal{T}(\mathrm{S},\mathrm{V})_{\mathcal{C}}$, we get the contradictory $c_0\psi_1=c_1\psi_1$ by Corollaries 2.8 and 3.8. In case of $T\in\{\mathcal{BT}(\mathrm{S,V}),\mathcal{EBT}(\mathrm{S,V})\}$, we know that c_0, c_1 are constructor-equal by Corollary 3.12. By Corollary 2.9 there is some $\mu:\mathcal{V}(c_0,c_1)\to\mathcal{T}(\mathrm{S},\mathrm{V})$ with $c_0\mu=c_1\mu$, which contradicts the case again.

Case 4: In this case we have $P_i(y_{i,1},\ldots,y_{i,m'_i})\sigma$ in D'_{k_i}. Then, for all $i\in\{0,1\}$, there are $I_i\subseteq\{1,\ldots,m'_i\}$ and, for all $j\in I_i$, there are $\hat{v}_{i,j}\in\mathcal{BT}(\mathrm{S,V})$ and $\varrho_{i,j}:\mathcal{V}(\hat{v}_{i,j})\to\mathcal{T}(\mathrm{S},\mathrm{V})$ with $y_{i,j}\sigma\psi_1\overset{*}{\longrightarrow}_{n_{k_i}}\hat{v}_{i,j}\varrho_{i,j}$, such that $P_i(\hat{v}_{i,1},\ldots,\hat{v}_{i,m'_i})$ is known to hold for all $\hat{v}_{i,j}\in\mathcal{BT}(\mathrm{S,V})$ $(j\in\{1,\ldots,m'_i\}\backslash I_i)$.
Thus, by the known theorem on P_0 and P_1, namely

$$\neg\exists y_{0,1},\ldots,y_{0,m'_0},y_{1,1},\ldots,y_{1,m'_1}\in\mathcal{BT}(\mathrm{S,V}).\left(\begin{array}{rl} & P_0(y_{0,1},\ldots,y_{0,m'_0})\\ \wedge & P_1(y_{1,1},\ldots,y_{1,m'_1})\end{array}\right),$$

as the variables $y_{i,1},\ldots,y_{i,m'_i}$ are distinct, there must be $n'_0\in I_0$ and $n'_1\in I_1$ with $y_{0,n'_0}=y_{1,n'_1}$, but $\hat{v}_{0,n'_0},\hat{v}_{1,n'_1}$ are not constant-equal. Thus, we have

$$\hat{v}_{0,n'_0}\varrho_{0,n'_0}\overset{*}{\longleftarrow}_{n_{k_0}}y_{0,n'_0}\sigma\psi_1=y_{1,n'_1}\sigma\psi_1\overset{*}{\longrightarrow}_{n_{k_1}}\hat{v}_{1,n'_1}\varrho_{1,n'_1}.$$

By shallow confluence and Corollary 3.2 we get $\hat{v}_{0,n'_0}\varrho_{0,n'_0}{\downarrow}\hat{v}_{1,n'_1}\varrho_{1,n'_1}$. Thus, by Corollary 3.12 get the contradictory fact that $\hat{v}_{0,n'_0},\hat{v}_{1,n'_1}$ are constant-equal.

Case 5: In this case we have $(y_{0,1}{=}y_{0,2})\sigma$ in D'_{k_0} and $P(y_{1,1},\ldots,y_{1,m'_1})\sigma$ in D'_{k_1} and there is an $l\in\{1,2\}$ such that (Def $y_{0,l}\sigma$) is in D'_{k_0} unless $y_{0,l}\sigma\in\mathcal{T}_{\mathcal{C}}(\mathrm{V})$. By Corollaries 2.8 and 3.8, we get $y_{0,l}\sigma\psi_1\overset{*}{\longrightarrow}_{n_{k_0}}\hat{u}\notin\mathrm{dom}(\longrightarrow)$ for some $\hat{u}$.
There is a $I\subseteq\{1,\ldots,m'_1\}$ and, for all $j\in I$, there are $\hat{v}_j\in\mathcal{BT}(\mathrm{S,V})$ and $\varrho_j:\mathcal{V}(\hat{v}_j)\to\mathcal{T}(\mathrm{S},\mathrm{V})$ with $y_{1,j}\sigma\psi_1\overset{*}{\longrightarrow}_{n_{k_1}}\hat{v}_j\varrho_j$, such that $P(\hat{v}_1,\ldots,\hat{v}_{m'_1})$ is known to hold for all $\hat{v}_j\in\mathcal{BT}(\mathrm{S,V})$ $(j\in\{1,\ldots,m'_1\}\backslash I)$.
Thus, by the known theorem on P, namely

$$\neg\exists y_{0,1},y_{0,2},y_{1,1},\ldots,y_{1,m'_1}\in\mathcal{BT}(\mathrm{S,V}).\left(\begin{array}{rl} & (y_{0,1}{=}y_{0,2})\\ \wedge & P(y_{1,1},\ldots,y_{1,m'_1})\end{array}\right),$$

as the variables $y_{1,1},\ldots,y_{1,m'_1}$ are distinct, there must be $n'_0,n'_1\in I$ with $y_{0,1}=y_{1,n'_0}$ and $y_{0,2}=y_{1,n'_1}$, but $\hat{v}_{n'_0},\hat{v}_{n'_1}$ are not constant-equal. Thus, we have

$$\hat{v}_{n'_0}\varrho_{n'_0}\overset{*}{\longleftarrow}_{n_{k_1}}y_{0,1}\sigma\psi_1\overset{*}{\longrightarrow}_{n_{k_0}}\circ\overset{*}{\longleftarrow}_{n_{k_0}}y_{0,2}\sigma\psi_1\overset{*}{\longrightarrow}_{n_{k_1}}\hat{v}_{n'_1}\varrho_{n'_1}.$$

By [Wirth, 2008, Lemma 15(4)] and Corollaries 3.2 and 3.8, we get $\hat{v}_{n'_0}\varrho_{n'_0}{\downarrow}_{n_{k_0}}\hat{v}_{n'_1}\varrho_{n'_1}$. By Corollaries 3.2 and 3.12 we get the contradictory fact that $\hat{v}_{n'_0},\hat{v}_{n'_1}$ are constant-equal.

Q.e.d. (Lemma 3.15)

4 Specification

Definition 4.1 (Specification)

A *specification* $\mathrm{S} = (\mathbb{S}, \mathbb{F}, \mathbb{D}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{P}, \mathrm{R})$ is a 9-tuple with the following properties:

1. $(\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{F}, \mathbb{P}, \mathbb{D})$ is a signature. We re-use our notation on signatures — such as $\mathcal{T}(\mathrm{S}, \mathrm{V})$ — for specifications by considering the triple of the first seven elements of S as the signature referred to.

2. $\mathbb{P}$ is a set of *(constant) predicate symbols* disjoint from $\mathbb{F}$, such that, for each predicate symbol $P \in \mathbb{P}$, there is a declaration in $\mathbb{D}$ of the form $P : s_0, \ldots, s_{n-1}$ with $n \in \mathbf{N}$ and $s_j \in \mathbb{S}$ for $j \in \{0, \ldots, n{-}1\}$.

3. R is a CTRS, i.e. a conditional term rewriting system, i.e. a set of (conditional) equations. A *(conditional) equation* is a triple $l{=}r{\longleftarrow}C$ consisting of two terms of the same sort, namely the *left-hand side* l and the *right-hand side* r, and the *condition* C.[6] If the condition is empty, i.e. if $C = \emptyset$, the equation is *unconditional* and we simply write $l{=}r$ instead of $l{=}r{\longleftarrow}C$.

4. Let V be a variable-system for S. For each $s' \in \mathbb{S}_{\mathcal{NT}}$, we require that the set $\mathcal{CT}(\mathrm{S},\mathrm{V})_{s'}$ is non-empty.

5. All elements of $\mathbb{S}_{\mathcal{E}}$ are of the form $h{\multimap}v$ for some sorts $h, v \in \mathbb{S}$ with $(h{\multimap}v) > h > v$. And if $(h{\multimap}v) \in \mathbb{S}_{\mathcal{E}}$, then there is some $\_[\_] \in \mathbb{F}$ such that $\mathbb{D}$ contains the declaration $\_[\_] : (h{\multimap}v) \to h \to v,\ \mathcal{G}$.

6. For each declaration of the form
$$d : s_0 \to \cdots \to s_n,\ \mathcal{D}, j,$$
we require that $\mathbb{D}$ contains the following declarations:
   (a) In case of $s_j > s_n$, for $\hat{s} := s_n$:
   $$d^j_{s_0,\ldots,s_{n-1}} : s_0 \to \cdots \to s_{j-1} \to s_{j+1} \to \cdots \to s_{n-1} \to (s_j{\multimap}\hat{s}),\ \mathcal{C}.$$
   (b) In case of $s_j \sim s_n$, for any $\hat{s} \in \mathbb{S}$ with $(s_n{\multimap}\hat{s}) \in \mathbb{S}_{\mathcal{E}}$:
   $$\_d^j_{s_0,\ldots,s_{n-1}} : (s_n{\multimap}\hat{s}) \to s_0 \to \cdots \to s_{j-1} \to s_{j+1} \to \cdots \to s_{n-1} \to (s_j{\multimap}\hat{s}),\ \mathcal{C}.$$

   Moreover, the declarations of constructors of all experimental sorts $s_j{\multimap}\hat{s}$ are minimally generated this way by some destructor of s_j, and the set $\mathbb{S}_{\mathcal{E}}$ is the smallest set such that the sorts of the declarations of these constructor symbols belong to $\mathbb{S}$.

   Finally, R has to contain the following unconditional equations for some $X_k \in \mathrm{V}_{\mathcal{G},s_k}$ for $k \in \{0, \ldots, n{-}1\}$:
   (a) In case of $s_j > s_n$, for $\hat{s} := s_n$:
   $$(d^j_{s_0,\ldots,s_{n-1}}{:}\hat{s}) \quad d^j_{s_0,\ldots,s_{n-1}}(X_0, \ldots, X_{j-1}, X_{j+1}, \ldots, X_{n-1})[X_j] = d(X_0, \ldots, X_n)$$
   (b) In case of $s_j \sim s_n$, for any $\hat{s} \in \mathbb{S}$ with $(s_n{\multimap}\hat{s}) \in \mathbb{S}_{\mathcal{E}}$, and for $E \in \mathrm{V}_{\mathcal{G},s_n\multimap\hat{s}}$:
   $$(d^j_{s_0,\ldots,s_{n-1}}{:}\hat{s}) \quad E\, d^j_{s_0,\ldots,s_{n-1}}(X_0, \ldots, X_{j-1}, X_{j+1}, \ldots, X_{n-1})[X_j] = E[d(X_0, \ldots, X_n)]$$

[6] It is to be interpreted as: "For all variables from $\mathcal{V}(l{=}r{\longleftarrow}C)$, the left-hand side l is equal to the right-hand side r if the condition C holds."

Note that Items 5 and 6 of Definition 2.1 describe a generic way of extending the specification for the visible and hidden sorts that are smaller than the experimental sorts. Indeed, in [Roşu, 2004, Definition 4.1], this generic way of extension is defined as an afterthought in what is called "unhiding a signature". The reason why we have chosen to integrate it directly into our definition of a signature is that otherwise we cannot introduce greater sorts on the basis of the experimental sorts. This does not mean that we have to specify the experimental sorts explicitly in our examples. We can use them, however, for the specification of additional sorts, such as lists or streams of experiments.

Moreover, note that — contrary to [Roşu, 2004] — we may have experimental sorts $h \twoheadrightarrow \hat{s}$ with $\hat{s} \in \mathbb{S}_{\mathcal{D}}$, i.e. the results of experiments may be hidden sorts. Note that this may happen only if $h > \hat{s}$; but especially not if $h \sim \hat{s}$, in which case we end up in Item 9b of Definition 2.1 analogous to [Roşu, 2004, Definition 4.1].

Definition 4.2 (Satisfaction)

Let $\mathcal{S}$ be an S-structure for the specification $\mathrm{S} = (\mathbb{S}, \mathbb{F}, \mathbb{D}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \precsim, \mathbb{P}, \mathrm{R})$, i.e. for the signature $(\mathbb{S}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \precsim, \mathbb{F}, \mathbb{P}, \mathbb{D})$.

$\mathcal{S}$ is a *model of* S if the following holds:

- The constructor universes $(\mathcal{C}, s)^{\mathcal{S}}$ are constructed recursively over the well-founded ordering $<$ of the quasi-ordering $\precsim$ as follows:
  - If $s \in \mathbb{S}_{\mathcal{NT}} \cap \mathbb{S}_{\mathcal{NC}}$, then $(\mathcal{C}, s)^{\mathcal{S}} = \{\, t\sigma \mid t \in \mathcal{CT}(\mathrm{S,V}) \wedge \sigma : \mathcal{V}(t) \to \mathcal{S} \,\}$; and for any declaration of the form $c : s'_0 \to \cdots s'_{n-1} \to s,\ \mathcal{C}$ in $\mathbb{D}$ we have
  $$c^{\mathcal{S}}_{s'_0,\ldots,s'_{n-1}}(o_0, \ldots, o_{n-1}) = c_{s'_0,\ldots,s'_{n-1}}(o_0, \ldots, o_{n-1}).$$
  - If $s \in \mathbb{S}_{\mathcal{NT}} \backslash \mathbb{S}_{\mathcal{NC}}$, then $(\mathcal{C}, s)^{\mathcal{S}} = \{\, \langle\!\{t\sigma\}\!\rangle_{\approx} \mid t \in \mathcal{CT}(\mathrm{S,V}) \wedge \sigma : \mathcal{V}(t) \to \mathcal{S} \,\}$; and for any declaration of the form $c : s'_0 \to \cdots s'_{n-1} \to s,\ \mathcal{C}$ in $\mathbb{D}$ we have
  $$c^{\mathcal{S}}_{s'_0,\ldots,s'_{n-1}}(o_0, \ldots, o_{n-1}) = c_{s'_0,\ldots,s'_{n-1}}(o_0, \ldots, o_{n-1}).$$
  - If $s \in \mathbb{S}_{\mathcal{NC}} \backslash \mathbb{S}_{\mathcal{NT}}$,
  - If $s \in \mathbb{S} \backslash (\mathbb{S}_{\mathcal{NT}} \cup \mathbb{S}_{\mathcal{NC}})$,
  - if $s \in \mathbb{S}_{\mathcal{C}}$, for each $s' \in \mathbb{S}_{\mathcal{C}}$ with $s' \sim s$, we have that $(\mathcal{C}, s')^{\mathcal{S}}$ is the smallest set such that, for each $c \in \mathbb{C}$ declared as $c : s_0 \to \cdots \to s_{n-1} \to s'$, and for each $o_i \in (\mathcal{C}, s_i)^{\mathcal{S}}$ $(i \in \{0, \ldots, n{-}1\})$, we have $c(o_0, \ldots, o_{n-1}) \in (\mathcal{C}, s')^{\mathcal{S}}$; and,
  - if $s \in \mathbb{S}_{\mathcal{D}}$, then $(\mathcal{C}, s)^{\mathcal{S}}$ is the set of functions f whose domain is the set of pairs (e, τ), where $e \in E_s$ is an experiment of sort s, where τ is an $\mathcal{S}$-valuation on $\mathcal{V}(e) \setminus \{\bullet_{s,0}\}$, and where $f(e, \tau) \in (\mathcal{C}, s)^{\mathcal{S}}$.
- For each $c \in \Sigma_{\mathcal{C}}$ declared as $c : s_0 \to \cdots \to s_n$ we have the following:
  - In case of $s_n \in \Sigma_{\mathcal{C}}$:
  For each $o_0, \ldots, o_{n-1}$ with $o_i \in (\mathcal{C}, s_i)$ for each $i \in \{0, \ldots, n{-}1\}$, we have
  $$c^{\mathcal{S}}(o_0, \ldots, o_{n-1}) \;=\; c(o_0, \ldots, o_{n-1}).$$
- For each $d \in \mathbb{F}$ declared as $d : s_0 \to \cdots \to s_n, \mathcal{D}$ we have the following:

- In case of $s_n{<}s_0$:
  For each $o_0, \ldots, o_{n-1}$ with $o_i \in (\mathcal{C}, s_i)$ for each $i \in \{0, \ldots, n{-}1\}$, we have
  $$d^{\mathcal{S}}(o_0, \ldots, o_{n-1}) \;=\; o_0(e, \tau)$$
  for
  $$e \;:=\; d(\bullet_{s_0,0}, \bullet_{s_1,1}, \ldots, \bullet_{s_{n-1},n-1})$$
  and
  $$\tau \;:=\; \{\; \bullet_{s_j,j} \mapsto o_j \;|\; j \in \{1, \ldots, n{-}1\} \;\}.$$
- In case of $s_n{\sim}s_0$:
  For each $o_0, \ldots, o_{n-1}$ with $o_i \in (\mathcal{C}, s_i)$ for each $i \in \{0, \ldots, n{-}1\}$, we have
  $$d^{\mathcal{S}}(o_0, \ldots, o_{n-1}) \;=\; f$$
  where the function f is given as follows:
  For each $e \in E_{s_n}$ and each S-valuation τ on $\mathcal{V}(e) \setminus \{\bullet_{s,0}\}$, we have
  $$f(e, \tau) \;:=\; o_0(e\sigma, \tau{\uplus}\tau')$$
  for
  $$\sigma \;:=\; \{\; \bullet_{s_n,0} \mapsto d(\bullet_{s_0,0}, \bullet_{s_1,m+1}, \ldots, \bullet_{s_{n-1},m+n-1}) \;\}$$
  for
  $$m \;:=\; |\mathcal{V}(e)| - 1,$$
  and
  $$\tau' \;:=\; \{\; \bullet_{s_{m+j},j} \mapsto o_j \;|\; j \in \{1, \ldots, n{-}1\} \;\}.$$

Example 4.3 (Cf. [Wirth, 2008, Example 8])

Let $\mathrm{S} = (\mathbb{S}, \mathbb{F}, \mathbb{D}, \mathbb{S}_{\mathcal{NT}}, \mathbb{S}_{\mathcal{NC}}, \mathbb{S}_{\mathcal{E}}, \lesssim, \mathbb{P}, \mathrm{R})$ be generated from the following:

$\mathbb{S}_{\mathcal{C}} \ni \mathsf{bool},\ \ \mathsf{nat},\ \ (\mathsf{nat}, \mathsf{nat}),\ \ \mathsf{list}(\mathsf{term}),\ \ \mathsf{term},\ \ (\mathsf{nat}, \mathsf{set}(\mathsf{term}))$

$\mathbb{S}_{\mathcal{D}} \ni \mathsf{set}(\mathsf{term}),\ \ \mathsf{stream}((\mathsf{nat}, \mathsf{set}(\mathsf{term})))$

$\mathsf{nat} \lesssim (\mathsf{nat}, \mathsf{nat})$
$\mathsf{nat} \lesssim \mathsf{term} \lesssim \mathsf{list}(\mathsf{term}) \lesssim \mathsf{term}$
$\mathsf{term} \lesssim \mathsf{set}(\mathsf{term}) \lesssim (\mathsf{nat}, \mathsf{set}(\mathsf{term})) \lesssim \mathsf{stream}((\mathsf{nat}, \mathsf{set}(\mathsf{term})))$

$\mathbb{F} \ni \mathsf{true},\ \ \mathsf{false},\ \ \mathsf{0},\ \ \mathsf{s},\ \ (\_,\_),\ \ \mathsf{nil},\ \ \mathsf{cons},\ \ \mathsf{var},\ \ \mathsf{term},\ \ \mathsf{set}$

$$\mathbb{D} \ni \left\{ \begin{array}{rcrcll} \mathsf{true},\ \ \mathsf{false} & : & & & \mathsf{bool}, & \mathcal{C} \\ \mathsf{0} & : & & & \mathsf{nat}, & \mathcal{C} \\ \mathsf{s} & : & \mathsf{nat} & \rightarrow & \mathsf{nat}, & \mathcal{C} \\ (\_,\_) & : & \mathsf{nat} \rightarrow \mathsf{nat} & \rightarrow & (\mathsf{nat}, \mathsf{nat})\ , & \mathcal{C} \\ \mathsf{nil} & : & & & \mathsf{list}(\mathsf{term}), & \mathcal{C} \\ \mathsf{cons} & : & \mathsf{term} \rightarrow \mathsf{list}(\mathsf{term}) & \rightarrow & \mathsf{list}(\mathsf{term}), & \mathcal{C} \\ \mathsf{var} & : & \mathsf{nat} & \rightarrow & \mathsf{term}, & \mathcal{C} \\ \mathsf{term} & : & \mathsf{nat} \rightarrow \mathsf{list}(\mathsf{term}) & \rightarrow & \mathsf{term}, & \mathcal{C} \\ \mathsf{set} & : & \mathsf{list}(\mathsf{term}) & \rightarrow & \mathsf{set}(\mathsf{term}), & \mathcal{C} \\ (\_,\_) & : & \mathsf{nat} \rightarrow \mathsf{set}(\mathsf{term}) & \rightarrow & (\mathsf{nat}, \mathsf{set}(\mathsf{term})), & \mathcal{C} \end{array} \right.$$

The domains are the Boolean values bool given by the constructors true and false, the natural numbers nat given by the constructors zero $\mathsf{0}$ and successor s, pairs over natural numbers $(\mathsf{nat}, \mathsf{nat})$ given by the constructor $(\_,\_)$, lists of terms $\mathsf{list}(\mathsf{term})$ given by the constructors nil and cons, and terms term given by the constructors var (mapping a natural number to a variable) term (constructing a term from a natural number encoding a function symbol and a list of terms).

$\mathbb{F} \ni \mathsf{in},\ \ \mathsf{hd},\ \ \mathsf{tl}$

$$\mathbb{D} \ni \left\{ \begin{array}{rcrcll} \mathsf{in} & : & \mathsf{term} \rightarrow \mathsf{set}(\mathsf{term}) & \rightarrow & \mathsf{bool}, & \mathcal{D} \\ \mathsf{hd} & : & \mathsf{stream}((\mathsf{nat}, \mathsf{set}(\mathsf{term}))) & \rightarrow & (\mathsf{nat}, \mathsf{set}(\mathsf{term})), & \mathcal{D} \\ \mathsf{tl} & : & \mathsf{stream}((\mathsf{nat}, \mathsf{set}(\mathsf{term}))) & \rightarrow & \mathsf{stream}((\mathsf{nat}, \mathsf{set}(\mathsf{term}))), & \mathcal{D} \end{array} \right.$$

The domains are

$\mathbb{F} \ni \_+\_,\ \ \_-\_,\ \ \mathsf{divrest},\ \ \mathsf{well},\ \ \mathsf{welltail},\ \ \mathsf{length},\ \ \mathsf{mbp},\ \ \mathsf{or},\ \ \mathsf{and},\ \ \mathsf{arity}$

$$\mathbb{D} \ni \left\{ \begin{array}{rcrcll} \mathsf{or},\ \ \mathsf{and} & : & \mathsf{bool} \rightarrow \mathsf{bool} & \rightarrow & \mathsf{bool}, & \mathcal{G} \\ \mathsf{well} & : & \mathsf{term} & \rightarrow & \mathsf{bool}, & \mathcal{G} \\ \mathsf{welltail} & : & \mathsf{list}(\mathsf{term}) & \rightarrow & \mathsf{bool}, & \mathcal{G} \\ \mathsf{mbp} & : & \mathsf{term} \rightarrow \mathsf{list}(\mathsf{term}) & \rightarrow & \mathsf{bool}, & \mathcal{G} \\ \mathsf{arity} & : & \mathsf{nat} & \rightarrow & \mathsf{nat}, & \mathcal{G} \\ \_+\_,\ \ \_-\_ & : & \mathsf{nat} \rightarrow \mathsf{nat} & \rightarrow & \mathsf{nat}, & \mathcal{G} \\ \mathsf{length} & : & \mathsf{list}(\mathsf{term}) & \rightarrow & \mathsf{nat}, & \mathcal{G} \\ \mathsf{divrest} & : & \mathsf{nat} \rightarrow \mathsf{nat} & \rightarrow & (\mathsf{nat}, \mathsf{nat}), & \mathcal{G} \end{array} \right.$$

$\mathbb{P} \ni \_<\_,\ \ \_\leq\_$

$\mathbb{D} \ni \{\ \_<\_,\ \ \_\leq\_\ \ :\ \ \mathsf{nat}, \mathsf{nat}$

$\mathrm{V}_{\mathcal{C},\mathsf{nat}} \ni x, y, z_1, z_2, r, q$

$\mathrm{V}_{\mathcal{G},\mathsf{bool}} \ni B_1, B_2$

$\mathrm{V}_{\mathcal{G},\mathsf{term}} \ni T$

$\mathrm{V}_{\mathcal{G},\mathsf{list(term)}} \ni L$

Let $\mathrm{R}_{4.3}$ denote the CTRS consisting of the following (conditional) equations:

$$\begin{array}{ll}
(\mathsf{in:set1}) & \mathsf{in}(T,\mathsf{set}(L)) = \mathsf{mbp}(T,L)
\end{array}$$

$$\begin{array}{lll|lll}
(\mathsf{plus1}) & x+\mathsf{0} & = x & (\mathsf{minus1}) & x-\mathsf{0} & = x \\
(\mathsf{plus2}) & x+\mathsf{s}(y) & = \mathsf{s}(x+y) & (\mathsf{minus2}) & \mathsf{s}(x)-\mathsf{s}(y) & = x-y
\end{array}$$

$$\begin{array}{llll}
(\mathsf{divrest1}) & \mathsf{divrest}(x,y) = (0,x) & \longleftarrow & x<y \\
(\mathsf{divrest2}) & \mathsf{divrest}(x,y) = (\mathsf{s}(q),r) & \longleftarrow & y\leq x,\ \ y\neq \mathsf{0},\ \ \mathsf{divrest}(x-y,y)=(q,r)
\end{array}$$

$$\begin{array}{lllll}
(\mathsf{well1}) & \mathsf{well}(\mathsf{var}(x)) & = \mathsf{true} & & \\
(\mathsf{well2}) & \mathsf{well}(\mathsf{term}(x,L)) & = \mathsf{false} & \longleftarrow & \mathsf{arity}(x)=z_1,\ \ \mathsf{length}(L)=z_2,\ \ z_1\neq z_2 \\
(\mathsf{well3}) & \mathsf{well}(\mathsf{term}(x,L)) & = \mathsf{welltail}(L) & \longleftarrow & \mathsf{arity}(x)=z_1,\ \ \mathsf{length}(L)=z_2,\ \ z_1=z_2
\end{array}$$

$$\begin{array}{lllll}
(\mathsf{welltail1}) & \mathsf{welltail}(\mathsf{nil}) & = \mathsf{true} & & \\
(\mathsf{welltail2}) & \mathsf{welltail}(\mathsf{cons}(T,L)) & = \mathsf{false} & \longleftarrow & \mathsf{well}(T)=\mathsf{false} \\
(\mathsf{welltail3}) & \mathsf{welltail}(\mathsf{cons}(T,L)) & = \mathsf{welltail}(L) & \longleftarrow & \mathsf{well}(T)=\mathsf{true}
\end{array}$$

$$\begin{array}{lll|lll}
(\mathsf{length1}) & \mathsf{length}(\mathsf{nil}) & = \mathsf{0} & (\mathsf{mbp1}) & \mathsf{mbp}(T,\mathsf{nil}) & = \mathsf{false} \\
(\mathsf{length2}) & \mathsf{length}(\mathsf{cons}(T,L)) & = \mathsf{s}(\mathsf{length}(L)) & & & \\
(\mathsf{or1}) & \mathsf{or}(\mathsf{false},B_2) & = B_2 & (\mathsf{and1}) & \mathsf{and}(\mathsf{false},B_2) & = \mathsf{false} \\
(\mathsf{or2}) & \mathsf{or}(B_1,\mathsf{false}) & = B_1 & (\mathsf{and2}) & \mathsf{and}(B_1,\mathsf{false}) & = \mathsf{false} \\
(\mathsf{or3}) & \mathsf{or}(\mathsf{true},B_2) & = \mathsf{true} & (\mathsf{and3}) & \mathsf{and}(\mathsf{true},B_2) & = B_2 \\
(\mathsf{or4}) & \mathsf{or}(B_1,\mathsf{true}) & = \mathsf{true} & (\mathsf{and4}) & \mathsf{and}(B_1,\mathsf{true}) & = B_1
\end{array}$$

There are addition $\_+\_$, minus $\_-\_$, and a division function $\mathsf{divrest}(\_,\_)$. The function $\mathsf{divrest}$ computes a pair consisting of the quotient and the remainder. Moreover, there is a well-formedness predicate $\mathsf{well}(\_)$ (actually a characteristic function). $\mathsf{well}(\_)$ checks whether the number of arguments of each function symbol in a term is equal to its arity according to the partially specified function $\mathsf{arity}(\_)$. Furthermore, there is an incomplete definition of the member predicate $\mathsf{mbp}(\_,\_)$ on lists of terms $\mathsf{list(term)}$, which will be extended in what follows in different ways. Finally, there are definitions of Boolean disjunction $\mathsf{or}(\_,\_)$ and conjunction $\mathsf{and}(\_,\_)$, which are non-strict, i.e. which may return *defined* values (i.e. constructor terms) even when some arguments are undefined.

Notice that minus, $\mathsf{divrest}$, well, and mbp are *partial due to the specifying case analysis* (minus and mbp syntactically, the others by their conditions). There may also be functions that are partially defined due to non-termination. Partial and lazy incremental specification is important for inductive theorem proving in practice; cf. [Löchner, 2006] and [Wirth, 2006].

Furthermore, in the equation ($\mathsf{divrest2}$) we have an example of a *binding equation*, namely $\mathsf{divrest}(x-y,y)=(q,r)$, which introduces the *extra variables* q, r, which do not occur in the function-call $\mathsf{divrest}(x,y)$ of the left-hand side, but are then used to describe the return-value $(\mathsf{s}(q),r)$ of the right-hand side. In the equations ($\mathsf{well2}$) and ($\mathsf{well3}$) we find the binding equations $\mathsf{arity}(x)=z_1$ and $\mathsf{length}(L)=z_2$, binding the extra variables z_1 and z_2 for later use in the same condition list.

Finally note that the predicate symbols from $\mathbb{P}$ do not receive a specification by a CTRS. They are built-in predicates on the constructor sub-universes.

Let us see how the implicitly given experimental sorts and their constructors look like. To simplify notation let us write stream instead of $\mathsf{stream}((\mathsf{nat},\mathsf{set}(\mathsf{term})))$.

$$\mathbb{S}_{\mathcal{E}} \ni \mathsf{set}(\mathsf{term}){\multimap}\mathsf{bool}, \quad \mathsf{stream}{\multimap}(\mathsf{nat},\mathsf{set}(\mathsf{term}))$$

$$\mathrm{V}_{\mathcal{G},\mathsf{set}(\mathsf{term})} \ni M$$

$$\mathrm{V}_{\mathcal{G},\mathsf{stream}} \ni S$$

$$\mathrm{V}_{\mathcal{G},\mathsf{stream}\multimap(\mathsf{nat},\mathsf{set}(\mathsf{term}))} \ni E$$

$$\mathbb{D} \ni \left\{ \begin{array}{rcrcl} \mathsf{in}_1 & : & \mathsf{term} & \rightarrow & \mathsf{set}(\mathsf{term}){\multimap}\mathsf{bool} \\ \mathsf{hd}_0 & : & & & \mathsf{stream}{\multimap}(\mathsf{nat},\mathsf{set}(\mathsf{term})) \\ \_\ \mathsf{tl}_0 & : & \mathsf{stream}{\multimap}(\mathsf{nat},\mathsf{set}(\mathsf{term})) & \rightarrow & \mathsf{stream}{\multimap}(\mathsf{nat},\mathsf{set}(\mathsf{term})) \\ \mathsf{app} & : & \mathsf{set}(\mathsf{term}){\multimap}\mathsf{bool} \rightarrow \mathsf{set}(\mathsf{term}) & \rightarrow & \mathsf{bool} \\ \mathsf{app} & : & \mathsf{stream}{\multimap}(\mathsf{nat},\mathsf{set}(\mathsf{term})) \rightarrow \mathsf{stream} & \rightarrow & (\mathsf{nat},\mathsf{set}(\mathsf{term})) \end{array} \right.$$

$$\begin{array}{lll} (\mathsf{in}_1 : \mathsf{bool}) & \mathsf{in}_1(T)[M] & = \mathsf{in}(T,M) \\ (\mathsf{hd}_0 : (\mathsf{nat},\mathsf{set}(\mathsf{term}))) & \mathsf{hd}_0[S] & = \mathsf{hd}(S) \\ (\mathsf{tl}_0 : (\mathsf{nat},\mathsf{set}(\mathsf{term}))) & E\,\mathsf{tl}_0[S] & = E[\mathsf{tl}(S)] \end{array}$$

Example 4.4 Let the CTRS $\mathrm{R}_{4.4}$ consist of $\mathrm{R}_{4.3}$ of Example 4.3 plus:

$$\begin{array}{llcl} (\mathsf{mbp2}) & \mathsf{mbp}(p_1,\mathsf{cons}(p_2,L)) = \mathsf{true} & \longleftarrow & p_1{=}p_2 \\ (\mathsf{mbp3}) & \mathsf{mbp}(p_1,\mathsf{cons}(p_2,L)) = \mathsf{mbp}(p_1,L) & \longleftarrow & p_1{\neq}p_2 \end{array}$$

Together with (mbp1) from Example 4.3, the equations (mbp2) and (mbp3) provide us with a most natural definition of membership in lists. It is also one with the minimal number of defined function symbols. Note that the addition of the equations (mbp2) and (mbp3) constitutes a consistent extension of the specification of Example 4.3. We consider it to be obvious that the exclusion of this specification of membership in lists by admissibility conditions would constrict CTRSs to be a more or less theoretical enterprise. Cf. [Wirth, 2008, § 6] for further discussion on this.

Definition 4.5 (Experiment)

Let V be a variable-system for the signature $\mathrm{S} = (\mathbb{S},\mathbb{F},\mathbb{D},\mathbb{S}_{\mathcal{NT}},\mathbb{S}_{\mathcal{NC}},\mathbb{S}_{\mathcal{E}},\lesssim,\mathbb{P},\mathrm{R})$.

Let $\bullet : (\mathbb{S}{\times}\mathbf{N}) \rightarrow \mathrm{V}$ be an injective function such that $\bullet_{s,n} \in \mathrm{V}_{\mathcal{C},s}$ for all $s \in \mathbb{S}$ and $n \in \mathbf{N}$.

Let $\hat{s} \in \mathbb{S}_{\mathcal{D}}$.

Now the sets of *experiments* $E_{s'}$ for the sorts $s' \in \mathbb{S}_{\mathcal{D}}$ with $s'{\sim}\hat{s}$ are the smallest sets such that for each $d \in \mathbb{F}$ declared as $d : s' \rightarrow s_1 \rightarrow \cdots \rightarrow s_n, \mathcal{D}$ we have:

- In case of $s_n{<}s'$:
$$d(\bullet_{s',0},\bullet_{s_1,1},\ldots,\bullet_{s_{n-1},n-1}) \quad \in \quad E_{s'}.$$
- In case of $s_n{\sim}s'$, for each $e \in E_{s_n}$:
$$e\sigma \quad \in \quad E_{s'}$$
for
$$\sigma \;:=\; \{\; \bullet_{s_n,0} \mapsto d(\bullet_{s',0},\bullet_{s_1,m+1},\ldots,\bullet_{s_{n-1},m+n-1}) \;\}$$
for
$$m \;:=\; |\mathcal{V}(e)| - 1.$$

References

[Abramsky &al., 1992] S. Abramsky, Dov Gabbay, and T. S. E. Maibaum, editors. *Handbook of Logic in Computer Sci.*. Clarendon Press, 1992.

[Aït-Kaci & Nivat, 1989] Hassan Aït-Kaci and Maurice Nivat, editors. *Resolution of Equations in Algebraic Structures*. Academic Press (Elsevier), 1989.

[Avenhaus & Madlener, 1990] JürgenAvenhaus and Klaus Madlener. Term rewriting and equational reasoning. In R. B. Banerji, editor, *Formal Techniques in Artificial Intelligence*, pages 1–43. Elsevier, 1990.

[Bachmair & Ganzinger, 1991] Leo Bachmair and Harald Ganzinger. Perfect model semantics for logic programs with equality. 1991. In [Furukawa, 1991, pp. 645–659].

[Barendregt, 1984] Hen(dri)k P. Barendregt. *The Lambda Calculus*. North-Holland (Elsevier), 1984. 2nd ed..

[Becker, 1993] Klaus Becker. Semantics for Positive/Negative-Conditional Rewrite Systems. In *3rd Int. Workshop on Conditional Term Rewriting Systems, Pont-à-Mousson (France), 1992*, number 656 in Lecture Notes in Computer Science, pages 213–225. Springer, 1993.

[Bergstra & Klop, 1986] Jan A. Bergstra and Jan Willem Klop. Conditional rewrite rules: Confluence and termination. *J. Computer and System Sci.*, 32:323–362, 1986.

[Bidoit & Hennicker, 2006] Michel Bidoit and Rolf Hennicker. Constructor-based observational logic. *J. of Logic and Algebraic Programming*, 67:3–51, 2006.

[Courcelle, 1984] Bruno Courcelle, editor. *Proc. 9th Int. Colloquium on Trees in Algebra and Programming (CAAP 1984), Bordeaux*. Cambridge Univ. Press, 1984.

[Deemter & Peters, 1996] Kees van Deemter and Stanley Peters, editors. *Semantic Ambiguity and Underspecification*. CLSI LN No. 55, Stanford, 1996.

[Dershowitz & Jouannaud, 1990] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. 1990. In [Leeuwen, 1990, Vol. B, pp. 243–320].

[Furukawa, 1991] Koichi Furukawa, editor. *Proc. 8th Int. Conf. on Logic Programming*. MIT Press, 1991.

[Gentzen, 1938] Gerhard Gentzen. Die gegenwärtige Lage in der mathematischen Grundlagenforschung – Neue Fassung des Widerspruchsfreiheitsbeweises für die reine Zahlentheorie. *Forschungen zur Logik und zur Grundlegung der exakten Wissenschaften, Neue Folge*, 4:3–44, 1938. Reprinted as facsimile by Wissenschaftliche Buchgesellschaft, Darmstadt, Verlag von S. Hirzel, Stuttgart. English translation in [Gentzen, 1969].

[Gentzen, 1969] Gerhard Gentzen. *The Collected Papers of Gerhard Gentzen*. North-Holland (Elsevier), 1969. Ed. by Manfred E. Szabo.

[Gillman, 1987] Leonard Gillman. *Writing Mathematics Well*. The Mathematical Association of America, 1987.

[Gogolla, 1983] Martin Gogolla. *Algebraic Specifications with Partially Ordered Sorts and Declarations*. Report 169/1983, FB Informatik, Univ. Dortmund, 1983. Short version is [Gogolla, 1984].

[Gogolla, 1984] Martin Gogolla. Partially ordered sorts in algebraic specifications. 1984. In [Courcelle, 1984, pp. 139–154]. Long version is [Gogolla, 1983].

[han, 1993] Handbook of logic in artificial intelligence and logic programming. Vol. 1: Logical foundations, 1993.

[Hanus, 1994] Michaël Hanus. *On Extra Variables in (Equational) Logic Programming.* MPI–I–94–2–246, Max Planck Inst. für Informatik, Saarbrücken, 1994. Short version in [Sterling, 1991, pp. 665–679].

[Hanus, 2006] Michaël Hanus. CURRY — an integrated functional logic language. Web only: `http://www.informatik.uni-kiel.de/~curry/report.html`, 2006.

[Hobbs, 1996] Jerry R. Hobbs. Monotone decreasing quantifiers in a scope-free logical form. 1996. In [Deemter & Peters, 1996, pp. 55–76]. `http://www.isi.edu/~hobbs/monotone-decreasing.pdf`.

[Hudlak &al., 1999] Paul Hudlak, John Peterson, and Joseph H. Fasel. A gentle introduction to HASKELL. Web only: `http://www.haskell.org/tutorial`, 1999.

[Huet & Lankford, 1978] Gérard Huet and Dallas S. Lankford. *On the Uniform Halting Problem for Term Rewriting Systems.* Rapport de Recherche 283, INRIA, 1978. `http://www.ens-lyon.fr/LIP/REWRITING/OLD_PUBLICATIONS_ON_TERMINATION/HUET_LANKFORD`.

[Kaplan, 1988] Stéphane Kaplan. Positive/Negative-Conditional Rewriting. In *1st Int. Workshop on Conditional Term Rewriting Systems, Orsay (France), 1987*, number 308 in Lecture Notes in Computer Science, pages 129–143. Springer, 1988.

[Kühler & Wirth, 1996] Ulrich Kühler and Claus-Peter Wirth. *Conditional Equational Specifications of Data Types with Partial Operations for Inductive Theorem Proving.* SEKI-Report SR–1996–11 (ISSN 1437–4447). SEKI Publications, Univ. Kaiserslautern, 1996. Short version in 8th RTA 1997, LNCS 1232, pp. 38–52, Springer. `http://www.ags.uni-sb.de/~cp/p/rta97`.

[Klop, 1992] Jan Willem Klop. Term rewriting systems. 1992. In [Abramsky &al., 1992, Vol. II, pp. 1–116].

[Leeuwen, 1990] Jan van Leeuwen, editor. *Handbook of Theoretical Computer Sci.*. MIT Press, 1990.

[Löchner, 2006] Bernd Löchner. Things to know when implementing LPO. *Int. J. Artificial Intelligence Tools*, 15:53–79, 2006.

[Nipkow &al., 2000] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. ISABELLE's Logics: HOL. Web only: `http://isabelle.in.tum.de/PSV2000/doc/logics-HOL.pdf`, 2000.

[Nipkow &al., 2002] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. ISABELLE/HOL — *A Proof Assistant for Higher-Order Logic*. Number 2283 in Lecture Notes in Computer Science. Springer, 2002.

[Padawitz, 1988] Peter Padawitz. *Computing in Horn Clause Theories*. Springer, 1988.

[Padawitz, 2000] Peter Padawitz. Swinging types = functions + relations + transition systems. *Theoretical Computer Sci.*, 243:93–165, 2000.

[Plaisted, 1993] David Plaisted. Equational reasoning and term rewriting systems. 1993. In [han, 1993, pp. 273–364].

[Roşu, 2003] Grigore Roşu. Inductive behavioral proofs by unhiding. *Electronic Notes in Theoretical Computer Sci.*, 82:285–302, 2003. `http://dx.doi.org/10.1016/S1571-0661(04)80645-5`. Revised version is [Roşu, 2004].

[Roşu, 2004] Grigore Roşu. Behavioral abstraction is hiding information. *Theoretical Computer Sci.*, 327:197–221, 2004. `http://dx.doi.org/10.1016/j.tcs.2004.07.027`. Preliminary version is [Roşu, 2003].

[Slind, 1996] Konrad Slind. Function definition in higher-order logic. In *9th TPHOLs 1996*, number 1125 in Lecture Notes in Computer Science, pages 381–397. Springer, 1996.

[Smoka &al., 1989] Gerd Smoka, Werner Nutt, Joseph Goguen, and José Meseguer. Order-sorted equational computation. 1989. In [Aït-Kaci & Nivat, 1989, Vol. 2, pp. 297–367].

[Sterling, 1991] Leon Sterling, editor. *Proc. 12th Int. Conf. on Logic Programming*. MIT Press, 1991.

[Suzuki &al., 1995] Taro Suzuki, Aart Middeldorp, and Tetsuo Ida. Level-confluence of conditional rewrite systems with extra variables in right-hand sides. In *6th Int. Conf. on Rewriting Techniques and Applications, Kaiserslautern, 1995*, number 914 in Lecture Notes in Computer Science, pages 179–193. Springer, 1995.

[Waldmann, 1989a] Uwe Waldmann. *Semantics of Order-Sorted Specifications*. Report 297/1989, FB Informatik, Univ. Dortmund, 1989. Revised version in Theoretical Computer Sci. (1992) **94**, pp. 1–35, Elsevier.

[Waldmann, 1989b] Uwe Waldmann. *Unification in Order-Sorted Signatures*. Report 298/1989, FB Informatik, Univ. Dortmund, 1989. Revised version 1990, `http://www.mpi-sb.mpg.de/~uwe/paper/UNIDO-INF-298.dvi.gz`.

[Wirth, 1997] Claus-Peter Wirth. *Positive/Negative-Conditional Equations: A Constructor-Based Framework for Specification and Inductive Theorem Proving*, volume 31 of *Schriftenreihe Forschungsergebnisse zur Informatik*. Verlag Dr. Kovač, Hamburg, Arnoldstr. 49, D–27763 Hamburg, 1997. ISBN 3–86064–551–X, `www.ags.uni-sb.de/~cp/p/diss`.

[Wirth & Gramlich, 1994a] Claus-Peter Wirth and Bernhard Gramlich. A constructor-based approach for positive/negative-conditional equational specifications. *J. Symbolic Computation*, 17:51–90, 1994. `http://www.ags.uni-sb.de/~cp/p/jsc94`.

[Wirth & Gramlich, 1994b] Claus-Peter Wirth and Bernhard Gramlich. On notions of inductive validity for first-order equational clauses. In Alan Bundy, editor, *12th Int. Conf. on Automated Deduction, Nancy, 1994*, number 814 in Lecture Notes in Artificial Intelligence, pages 162–176. Springer, 1994. `www.ags.uni-sb.de/~cp/p/cade94`.

[Wirth & Lunde, 1994] Claus-Peter Wirth and Rüdiger Lunde. *Writing Positive/Negative-Conditional Equations Conveniently.* SEKI-Report SR–1994–04 (ISSN 1437–4447). SEKI Publications, Univ. Kaiserslautern, 1994. `http://www.ags.uni-sb.de/~cp/p/swp9404`.

[Wirth, 2006] Claus-Peter Wirth. *Progress in Computer-Assisted Inductive Theorem Proving by Human-Orientedness and Descente Infinie?* SEKI-Working-Paper SWP–2006–01 (ISSN 1860–5931). SEKI Publications, Saarland Univ., 2006. `www.ags.uni-sb.de/~cp/p/swp200601/welcome.html`.

[Wirth, 2008] Claus-Peter Wirth. Shallow confluence of conditional term rewriting systems. *J. Symbolic Computation*, 2008. `http://www.ags.uni-sb.de/~cp/p/shallow`.