Available online at www.sciencedirect.com

SCIENCE DIRECT®

Theoretical Computer Science 355 (2006) 215–227

Theoretical Computer Science

www.elsevier.com/locate/tcs

# Quantifier-free logic for nondeterministic theories

Yngve Lamo[a], Michał Walicki[b,*]

[a] *Faculty of Engineering, Bergen University College, 5020 Bergen, Norway*
[b] *Department of Informatics, University of Bergen, 5020 Bergen, Norway*

**Abstract**

We develop a quantifier-free logic for deriving consequences of multialgebraic theories. Multialgebras are used as models for nondeterminism in the context of algebraic specifications. They are many sorted algebras with *set-valued* operations. Formulae are sequents over atoms allowing one to state set-inclusion or identity of 1-element sets (determinacy). We introduce a sound and weakly complete Rasiowa–Sikorski (R–S) logic for proving multialgebraic tautologies. We then extend this system for proving consequences of specifications based on translation of finite theories into logical formulae. Finally, we show how such a translation may be avoided—introduction of the *specific cut* rules leads to a sound and strongly complete Gentzen system for proving directly consequences of specifications. Besides giving examples of the general techniques of R–S and the specific cut rules, we improve the earlier logics for multialgebras by providing means to handle empty carriers (as well as empty result-sets) without the use of quantifiers, and to derive consequences of theories without translation into another format and without using general cut.

*Keywords:* Multialgebra; Algebraic specification; Nondeterminism; Rasiowa-Sikorski technique; Strong completeness; Specific cut

## 0. Introduction

The institution [4] of multialgebras, $\mathcal{MA}$ [8], provides a powerful algebraic framework for specification—primarily, but not exclusively, of nondeterministic behavior [5,8,13,14]. A nondeterministic operation returns the set of all possible outcomes. Hence, operations are interpreted as functions from the carrier to the powerset of the carrier. The particular case of the empty result set gives straightforwardly a subinstitution of partial algebras [9]. The logic has two atoms: set inclusion $t \prec t'$ holds iff the interpretation of $t$ is included in the interpretation of $t'$, and element equality $t \doteq t'$ holds iff the terms $t$ and $t'$ return the same 1-element set—in particular, both are deterministic.

Formulae used for writing specifications are sequents over atomic equalities and inclusions. Our objective is to design a quantifier-free logic for deriving consequences of such specifications. First, using the technique of Rasiowa–Sikorski (R–S) from [11], we design a sound and complete system R–S. This system could be seen as a sublogic of the first-order logic for multialgebras given by Konikowska and Białasik in [3]. However, the language from [3] does not include the element equality $\doteq$. This predicate cannot be expressed in that language by a set of formulae without the use of explicit quantifiers, and this is related to the fact that to express (non)emptiness of the carrier, quantified formulae are needed. In their language $\exists x : x \prec x$ is needed to express nonemptiness of the carrier, which in our language, can be

[*] Corresponding author.
*E-mail addresses:* yla@hib.no (Y. Lamo), michal@ii.uib.no (M. Walicki).

0304-3975/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2006.01.010

expressed by the quantifier-free formula $x \doteq x$ (with only implicit universal quantification over possible assignments). Finally, and most significantly, the language from [3], unless extended to full first-order, is not expressive enough to state nonemptiness of any result set. Consequently, even the quantifier-free tautologies have all to take into account the possibility that any involved term may yield an empty result.

This not only yields fewer and less specific tautologies, but has also more practical consequences. Writing specifications one often wants to state that a term is deterministic. The axiom $f(x) \doteq f(x)$ states that the operation $f$ is a (total) function, [1] and such statements figure naturally in the formulae one wants to prove—preferably without the use of full first-order logic. Besides, there is the whole tradition of algebraic specifications based on equational axioms and equational reasoning. The element equality, present in $\mathcal{MA}$, makes comparison and embedding of other institutions to the institution of multialgebras simple and straightforward, without the use of quantifiers, [8].

Having given an example motivating the use of nondeterminism and introduced the basic notions from multialgebras in Section 1, we design a sound and weakly complete R–S system in Section 2. Following [11] (and also [1–3,6,7]) we also define a unique deduction strategy which can be used for implementing the logic. In Section 3, we address the issue of proving consequences of specifications. Specifications are sets of sequents and we want to derive their consequences, i.e., new sequents. We indicate the required translation schema and extend the R–S system with one rule needed for this purpose. Finally, in Section 4, we transform the obtained system to a sound and strongly complete Gentzen calculus GS, which is more user-friendly than the R–S system for proving theorems by hand. In order to handle proofs of consequences of theories without any intermediary translation of the involved sequents, we replace the axiom rule (as well as various rules for the logical connectives) by the specific cut rules, originating from [10]. We thus obtain a Gentzen system—without the general cut rule—for direct reasoning about specifications. Besides extension of the language with the useful predicate $\doteq$, we consider this result a significant improvement—by simplification—of the full first-order Gentzen system from [3]. We give only the general ideas of the proofs of central results—details can be found in [8].

## 1. Nondeterminism and multialgebras

Nondeterminism appears often as an undesirable side-effect of low-level factors (like relative speed of parallel processors, unpredictability of the environment, etc.) which affect program's run-time behavior but may be hard to bring under programmer's control. However, nondeterminism may be also viewed as a result of abstracting from various detailed choices. As a means of description and specification, nondeterministic operators supply a useful, and even necessary abstraction mechanisms which cannot be replaced by the assumed alternative of underspecification, [9].

**Example 1.** We give (the basic idea of) a specification, using conditional equations, of a topological sorting of a graph. We do not specify all the details of graph representation and other operations but notice only the need for a specification of sets which, except for the constant $\emptyset$, contains the constructor $+ : Set \times El \to Set$, adding an element to the set and satisfying the standard axioms which ensure that order of insertions and multiplicity of elements do not matter:

$$\begin{array}{ll} \text{S1.} & (S+x)+x = S+x, \\ \text{S2.} & (S+x)+y = (S+y)+x. \end{array}$$

Now, the standard topological sorting of a graph, $TS : Graph \to Seq$ is specified using an auxiliary operation, $TS'$, taking an additional argument—a sequence—which is initially empty (Axiom 1). $TS'(G, L)$ checks if the set of nodes with zero indegree in the (original input) graph $G$—with all the nodes occurring in $L$ removed—$Oin(G, L)$, is empty, in which case the sequence $L$ is returned (Axiom 2). Otherwise, the nonempty set has the form $S + x$, and so an element of this set, e.g. $x$, is appended at the end of the sequence $L$ and $TS'$ continues with this extended sequence $L \cdot x$.

$$\begin{array}{lr} 1. & TS(G) = TS'(G, \varepsilon), \\ 2. & Oin(G, L) = \emptyset \Rightarrow TS'(G, L) = L, \\ 3. & Oin(G, L) = S + x \Rightarrow TS'(G, L) = TS'(G, L \cdot x). \end{array}$$

[1] Function is a, possibly partial, deterministic operation, i.e., returning at most one result for each argument.

This should correspond to the intuitive understanding of the algorithm and may even look plausible. But the set axiom S2 intrudes on this specification and things go awry with Axiom 3: various possibilities of selecting the node from the set $Oin(G, L)$, and hence various resulting lists, will be identified. The plausible equality: $(\emptyset + c) + b = Oin(\begin{array}{ccc} & a & \\ \swarrow & & \searrow \\ b & & c \end{array}, \varepsilon \cdot a) = (\emptyset + b) + c$, will lead to a much less plausible one: $\varepsilon \cdot a \cdot b \cdot c = TS'(\begin{array}{ccc} & a & \\ \swarrow & & \searrow \\ b & & c \end{array}, \varepsilon \cdot a) = \varepsilon \cdot a \cdot c \cdot b$.

The problem is not merely with this particular specification which could be somehow fixed. In fact, using only deterministic operations one can hardly imagine capturing the intuition of the *algorithm* which describes really a *set* of possible results. An algorithm can be viewed as a nondeterministic program where some choices, being inessential for the correctness, have been left to the implementor. Very often, such choices concern the order of iteration over some structures which are most adequately—and abstractly—treated as sets. The conflict thus arises between the abstract understanding of a set as an unordered structure and actual iteration which induces some ordering. As the example illustrates, different orderings may lead to different sequences which, however, all are among the intended and admissible results. To obtain an adequate specification of topological sorting we must allow it to return different results depending on the choices from the subsequent sets $Oin(G, L)$. On the other hand, in order not to *overspecify* the problem, we should not assume any particular ordering of the involved sets (as would be the case if, for instance, these were represented as lists). The adequate notion of an iterator over sets fulfilling these requirements is, in fact, nondeterministic choice. We are thus led to allow nondeterministic operations as appropriate means of abstraction. We model such operations using multialgebras where operations may return sets of (possible) results and not only unique results. (We will return to an adequate formulation of the above specification once we have introduced multialgebraic specifications and models.)

Multialgebras are presented using the standard algebraic signatures: $\Sigma = (S, \Omega)$, where $S$ is a set of sort names and $\Omega$ a set of operation names with profiles sorted over $S$. Terms over a signature $\Sigma$ and a set $X$ of variables, $\mathcal{T}_{\Sigma,X}$, are defined in the usual way. Occasionally we write $t_s$ to indicate that the term $t$ has sort $s$, but we always assume that terms are well sorted and usually drop the subscript. The well-formed formulae, $\mathcal{F}_{\Sigma,X}$, are the boolean combinations of the atomic formulae: $t_s \prec t'_s$ (inclusions) and $t_s \doteq t'_s$ (element equalities), where $t_s, t'_s \in \mathcal{T}_{\Sigma,X}$.

**Definition 2.** A $\Sigma$-multialgebra $A$ is a pair $A = (|A|, \Omega^A)$, where $|A| = \{s^A : s \in S\}$ is a (possibly empty) carrier set for each $s \in S$, and $\Omega^A = \{f^A : f \in \Omega\}$ is a set-valued function for each $f \in \Omega$, i.e., for each $f : s_1 \times \cdots \times s_n \to s$ we have: $f^A : s_1^A \times \cdots \times s_n^A \to \mathcal{P}(s^A)$, where $\mathcal{P}(s^A)$ is the set of all subsets of $s^A$. Composition is defined by pointwise extension: $f^A(g^A(x)) = \bigcup_{y \in g^A(x)} f^A(y)$.

An operation is partial if it returns the empty set for some arguments, and it is nondeterministic if it returns more than one value for some arguments. A function is an operation that is neither partial nor nondeterministic.

**Definition 3.** Given a multialgebra $A$, an assignment $\alpha$ is a function $\alpha : X \to |A| \uplus \{\emptyset\}$, where $\alpha(x_s) = \emptyset \iff s^A = \emptyset$.
A $\Sigma$ structure $M = \langle A, \alpha \rangle$ is a $\Sigma$ multialgebra $A$ with an assignment $\alpha$.

So an assignment assigns an element—not a *set* of elements!—to each variable of a nonempty sort. A special consequence of this definition is the existence of assignments to variables of empty sort. Alternatively, an assignment could be a partial function, with domain being the variables of nonempty sorts. Our formulation gives that any nonground term with variables from empty sort will be empty, as multialgebraic operations are strict on the empty set. Another consequence, concerning expression of (non)emptiness of sorts, is commented below.

Given a structure $M$, all terms $t \in \mathcal{T}_{\Sigma,X}$ obtain a unique interpretation, denoted by $t^M$, which is defined in the standard way.

**Definition 4.** Satisfaction in a structure $M = \langle A, \alpha \rangle$ is defined by
(1) $M \models t \prec t' \iff t^M \subseteq t'^M$;
(2) $M \models t \doteq t' \iff t^M = \{e\} = t'^M$, for some $e \in |A|$[2];
(3) $M \models \neg\gamma \iff M \not\models \gamma$, and $M \models \gamma \vee \phi \iff M \models \gamma$ or $M \models \phi$.

[2] The distinction between one-element set $\{e\}$ and the element $e$ is here inessential.

The symbol $\mathcal{E}_s$ will abbreviate the formula stating emptiness of the carrier $s$:

$$\mathcal{E}_s \equiv \neg(x_s \doteq x_s); \quad \text{and} \quad \neg\mathcal{E}_s \equiv x_s \doteq x_s, \quad \text{for any } x_s \in X_s.$$

According to point 2 of Definition 4, an equality may hold only if the carrier is nonempty. Combined with the specific feature of the assignment to empty sort it implies that, for a given structure $M = \langle A, \alpha\rangle$:

$$M \models \mathcal{E}_s \iff M \models \neg(x_s \doteq x_s) \iff s^A = \emptyset \quad \text{and} \quad M \models \neg\mathcal{E}_s \iff s^A \neq \emptyset.$$

## 2. The R–S calculus

Before designing the system to reason from theories, we start with a quantifier-free deduction system, R–S, for multialgebraic tautologies. It illustrates a powerful way of designing deduction systems based on the semantical properties of the atoms, originally introduced by Rasiowa and Sikorski in [11] (an overview with examples can be found in [6,7]). Our presentation in this section is an adaptation and extension of a similar logic described in [3].

The system processes sets of formulae (clauses). However, it allows one also to define a specific deduction strategy in which such sets are considered as ordered *sequences* of formulae.

**Definition 5.** A structure $M = \langle A, \alpha\rangle$ satisfies a sequence of formulae $\Gamma = \gamma_1, \dots, \gamma_n$, written $M \models \Gamma$, iff $M \models \gamma_i$ for some $i$. (“,’’ is a meta-disjunction.)

An R–S rule has one of the following forms, where $\Gamma_i$ are sequences:

$$\frac{\Gamma_1}{\Gamma_2}, \quad \frac{\Gamma_1}{\Gamma_2 \mid \Gamma_3} \quad \text{or} \quad \frac{\Gamma_1}{\Gamma_2 \mid \Gamma_3 \mid \Gamma_4}.$$

Both sides of “ | ’’ have to hold for making an expression involving | true, hence it should be viewed as a meta-conjunction. The rules are invertible and one uses the strong notion of soundness: an (R–S) rule is sound when, for any structure $M$, $M$ satisfies the premise if and only if it satisfies the conclusion.

Particular sequences are singled out as *axiomatic*.

**Definition 6.** An *axiomatic* sequence is a sequence containing a formula or a subsequence of the form (ignoring the order):

$$\text{(I)}\ x \prec x : x \in X, \quad \text{(II)}\ \phi, \neg\phi : \phi \in \mathcal{F}_{\Sigma,X} \quad \text{and} \quad \text{(III)}\ \neg\mathcal{E}_s, t_s \prec t'_s.$$

In addition to axiomatic sequences, one also identifies the *indecomposable* (formulae and) sequences. No rule can modify an indecomposable formula and so if such a formula appears during the proof, it will stay unchanged for the rest of the proof. In our case, these are given by the following definition.

**Definition 7.** A $\Sigma$, $X$ formula is indecomposable iff it has one of the forms:
- $\mathcal{E}_s$ or $\neg\mathcal{E}_s$, for $s \in S$
- $x \prec y$ or $\neg(x \prec y)$, where $x, y \in X$
- $x \prec f(x_1, \dots, x_n)$ or $\neg(x \prec f(x_1, \dots, x_n))$, where $x, x_i \in X$ and $f \in \Omega$.

A sequence $\Gamma$ is indecomposable iff every formula in $\Gamma$ is indecomposable.

The R–S calculus has two types of rules: for *replacement* and *expansion*. The replacement rules transform decomposable formulae leading either to axiomatic sequences or indecomposable formulae. They have only one explicit formula in the premise which is transformed, possibly with addition of a new formula. There is exactly one decomposition rule for each case of a decomposable formula and precisely one decomposition rule can be applied to any decomposable formula at any stage. In particular, we have one rule for every positive decomposable formula, like $t \doteq t'$, and one for its negative version, $\neg(t \doteq t')$.

The expansion rules are used to add logical consequences of the indecomposable formulae from the premise. They merely augment the premise sequence with such consequences without changing the formula itself.

The sign "*'' in the conclusion of a rule indicates repetition of the active formula from the premise. We have the following R–S proof system:

*Axiomatic sequences*: *containing* (*order does not matter*)

(I) $x \prec x \;\; : x \in X$, (II) $\gamma, \neg\gamma \;\; : \gamma \in \mathcal{F}_{\Sigma,X} \cup \{\mathcal{E}\}$ and (III) $\neg\mathcal{E}_s, t_s \prec t'_s$ .

*Replacement rules* (*unique decomposable premise formula*)

(IV) $$\frac{\Gamma', \neg\neg\gamma, \Gamma''}{\Gamma', \gamma, \Gamma''},$$

(V) $$\frac{\Gamma', \gamma \vee \phi, \Gamma''}{\Gamma', \gamma, \phi, \Gamma''}, \qquad \frac{\Gamma', \neg(\gamma \vee \phi), \Gamma''}{\Gamma', \neg\gamma, \Gamma'' \mid \Gamma', \neg\phi, \Gamma''},$$

(VI) $$\frac{\Gamma', \gamma \wedge \phi, \Gamma''}{\Gamma', \gamma, \Gamma'' \mid \Gamma', \phi, \Gamma''}, \qquad \frac{\Gamma', \neg(\gamma \wedge \phi), \Gamma''}{\Gamma', \neg\gamma, \neg\phi, \Gamma''},$$

(VII) $$\frac{\Gamma', t \prec t', \Gamma''}{\Gamma', \neg(x \prec t), x \prec t', \Gamma''}, \qquad \frac{\Gamma', \neg(t \prec t'), \Gamma''}{\Gamma', x \prec t, \Gamma'', * \mid \Gamma', \neg(x \prec t'), \Gamma'', *},$$

$t \notin X$ and $x \in X$ is fresh, $t \notin X$ and $x \in X$ arbitrary,

(VIII) $$\frac{\Gamma', x \prec f(..t..), \Gamma''}{\Gamma', y \prec t, \Gamma'', * \mid \Gamma', x \prec f(..y..), \Gamma'', *}, \qquad \frac{\Gamma', \neg(x \prec f(..t..)), \Gamma''}{\Gamma', \neg(y \prec t), \neg(x \prec f(..y..)), \Gamma''},$$

$y \in X$ arbitrary and $t \notin X$ , $y \in X$ is fresh and $t \notin X$ ,

(IX) $$\frac{\Gamma', t \doteq t', \Gamma''}{\Gamma', t \doteq x, \Gamma'', * \mid \Gamma', t' \doteq x, \Gamma'', *}, \qquad \frac{\Gamma', \neg(t \doteq t'), \Gamma''}{\Gamma', \neg(t \doteq x), \neg(t' \doteq x), \Gamma''},$$

$t, t' \notin X$ and $x \in X$ arbitrary, $t, t' \notin X$ and $x \in X$ is fresh,

(X) $$\frac{\Gamma', t_s \doteq x_s, \Gamma''}{\Gamma', t_s \prec x_s, \Gamma'' \mid \Gamma', x_s \prec t_s, \Gamma'' \mid \Gamma', \neg\mathcal{E}_s, \Gamma''}, \qquad \frac{\Gamma', \neg(t_s \doteq x_s), \Gamma''}{\Gamma', \mathcal{E}_s, \neg(x_s \prec t_s), \neg(t_s \prec x_s), \Gamma''},$$

$x_s \in X$ and $t_s \neq x_s$, $x_s \in X$ and $t_s \neq x_s$,

(XI) $$\frac{\Gamma', x_s \doteq t_s, \Gamma''}{\Gamma', t_s \prec x_s, \Gamma'' \mid \Gamma', x_s \prec t_s, \Gamma'' \mid \Gamma', \neg\mathcal{E}_s, \Gamma''}, \qquad \frac{\Gamma', \neg(x_s \doteq t_s), \Gamma''}{\Gamma', \mathcal{E}_s, \neg(x_s \prec t_s), \neg(t_s \prec x_s), \Gamma''},$$

$x_s \in X$ and $t_s \neq x_s$, $x_s \in X$ and $t_s \neq x_s$.

*Expansion rules* (*indecomposable premise formulae*):

(XII) $$\frac{\Gamma', \neg(x \prec y), \Gamma''}{\Gamma', \neg(x \prec y), \neg(y \prec x), \Gamma''}.$$

(XIII) $$\frac{\Gamma', \neg(y \prec x), \Gamma'', \neg(x \prec z), \Gamma'''}{\Gamma', \neg(y \prec x), \Gamma'', \neg(x \prec z), \neg(y \prec z), \Gamma'''}.$$

(XIV) $$\frac{\Gamma', \neg(y \prec x), \Gamma'', \neg(x \prec f(\overline{z})), \Gamma'''}{\Gamma', \neg(y \prec x), \Gamma'', \neg(x \prec f(\overline{z})), \neg(y \prec f(\overline{z})), \Gamma'''}.$$

(XV) $$\frac{\Gamma', \neg(y \prec z), \Gamma'', \neg(x \prec f(\ldots, z, \ldots)), \Gamma'''}{\Gamma', \neg(y \prec z), \Gamma'', \neg(x \prec f(\ldots, z, \ldots)), \neg(x \prec f(\ldots, y, \ldots)), \Gamma'''}.$$

(XVI) $$\frac{\Gamma', \neg\mathcal{E}_s, \Gamma'', \neg(x_{s'} \prec f(\ldots, y_s, \ldots)), \Gamma'''}{\Gamma', \neg\mathcal{E}_s, \Gamma'', \neg(x_{s'} \prec f(\ldots, y_s, \ldots)), \neg\mathcal{E}_{s'}, \Gamma'''}.$$

In the last three rules, $f$ may possibly be a constant.

## 3. Specifications and the system R–S*

The system R–S is sound and weakly complete and thus can be used to derive only tautologies (valid sequences) but not valid consequences of sets of axiomatic sequences. But, we are really interested in proving logical consequences of specifications, i.e., we need a strongly complete calculus. A specification $\mathcal{SP}$ is pair $(\Sigma, \Psi)$, where $\Sigma$ is a signature (kept implicit in what follows) and $\Psi$ is a set of axioms—sequents of atomic formulae. We now extend the R–S logic to fulfill this function by representing axiomatic sequents as sequences. In the following section, we will return to the sequent form and transform the obtained logic into a sound and strongly complete Gentzen system.

A $\Sigma$ sequent is a pair, written $\Gamma \Rightarrow \Delta$, of finite sets of formulae from $\mathcal{F}_{\Sigma,X}$. The notation means, implicitly, $\gamma_1, \ldots, \gamma_n \Rightarrow \delta_1, \ldots, \delta_m$. Following earlier works, e.g. [5,13,8], our specifications involve only sequents of *atomic* formulae (i.e., each $\gamma_i, \delta_j$ is either equality or inclusion), but we may occasionally need this more general definition. Keep also in mind that all formulae in a sequent are quantifier-free.

**Definition 8.** A sequent $\Gamma \Rightarrow \Delta$ is valid iff for every $M = \langle A, \alpha \rangle$ such that $M \models \gamma_i$, for all $\gamma_i \in \Gamma$, there exists a $\delta_j \in \Delta$ such that $M \models \delta_j$.

The function $\mathsf{tr}$ translates sequents to formulae in $\mathcal{F}_{\Sigma,X}$:
- $\mathsf{tr}(\gamma_1, \ldots, \gamma_n \Rightarrow \delta_1, \ldots, \delta_m) \equiv \neg\gamma_1 \vee \ldots \vee \neg\gamma_n \vee \delta_1 \vee \ldots \vee \delta_m$,
- for $\Psi = \{\psi_1, \ldots, \psi_n\} : \mathsf{tr}(\Psi) = \{\mathsf{tr}(\psi_1), \ldots, \mathsf{tr}(\psi_n)\}$.

The notation $\neg\Gamma, \Delta$ may be used for $\mathsf{tr}(\Gamma \Rightarrow \Delta)$ or, as the case may be, for the corresponding sequence $\neg\gamma_1, \ldots, \neg\gamma_n, \delta_1, \ldots, \delta_m$. The following lemma is straightforward.

**Lemma 9.** *For any structure M and sequent* $\psi : M \models \psi \iff M \models \mathsf{tr}(\psi)$. *In particular, a sequent* $\Gamma \Rightarrow \Delta$ *is valid iff the sequence* $\neg\Gamma, \Delta$ *is valid.*

The models for a specification are no longer structures with an assignment, but multialgebras satisfying the axioms for all possible assignments:

**Definition 10.** Given a specification $\mathcal{SP} = (\Sigma, \Psi)$, a $\Sigma$-multialgebra $A$, and a sequent $\psi$, we define the satisfaction relation $\models_*$:
(1) $A \models_* \psi \iff \forall\alpha.\langle A, \alpha \rangle \models \mathsf{tr}(\psi)$
(2) $A \models_* \Psi \iff \forall\psi_i \in \Psi. A \models_* \mathsf{tr}(\psi_i)$
(3) $\Psi \models_* \psi \iff \forall A.(A \models_* \Psi \Rightarrow A \models_* \mathsf{tr}(\psi))$.

The above definition may be applied also when $\psi$'s are arbitrary formulae or sequences, in which case we simply drop the applications of $\mathsf{tr}(\psi)$. This convention will be applied below—$\psi$ stands, in general, for arbitrary formula, while the notation $\mathsf{tr}(\psi)$ indicates that $\psi$ is a sequent.

Before giving the reasoning system for multialgebraic theories, we return to the specification of topological sorting and repair the problems from Example 1.

**Example 11.** We retain the specification of sets with the Axioms $S1$ and $S2$. The nondeterministic choice, $\sqcup : Set \to El$, can be specified, for instance as $x \prec \sqcup(S + x)$. The specification remains essentially the same but axioms must be adjusted to handle nondeterminism. The result of $TS(G)$ is no longer unique but may yield any result returned by the auxiliary operation, so in Axiom 1. $\doteq$ is replaced by $\prec$. Axiom 2 remains unchanged, and Axiom 3 is adjusted in a similar way as Axiom 1, with an application of nondeterministic choice from a nonempty set:

1. $TS(G) \prec TS'(G, \varepsilon)$,
3. $Oin(G, L) \doteq S + x \Rightarrow TS'(G, L) \prec TS'(G, L \cdot \sqcup(Oin(G, L)))$.

Instead of the problematic collapse of the sort of sequences from Example 1, we now obtain in any multialgebraic model $A$ of the specification the expected set of possible results: $TS'(\begin{smallmatrix} & a & \\ \swarrow & & \searrow \\ b & & c \end{smallmatrix}, \varepsilon \cdot a)^A \subseteq \{\varepsilon \cdot a \cdot b \cdot c^A, \varepsilon \cdot a \cdot c \cdot b^A\}$.

The condition in axiom 3 amounts only to ensuring nonemptiness of the set $Oin(G, L)$. The equality means that the argument to the choice in the consequent could be replaced with $S + x$. However, replacing the whole application $\sqcup(Oin(G, L))$ with $x$ would make the right-hand side of $\prec$ deterministic and result in the same collapse of distinct sequences as in the initial deterministic specification.

### 3.1. The system R–S*

An axiom $\psi$ is written $![\psi]$. We define its semantics (reflecting the intended $\models_*$), and extend the system R–S with a new rule to handle such formulae.

The procedure for extending the R–S system is quite standard—in order to prove a sequent $\psi$ from a finite specification $\Psi = \{\psi_1, \dots, \psi_n\}$, we perform a translation, $\mathsf{tr}$, of $\psi$ and all the sequents from $\Psi$ into formulae, form a sequence corresponding to $(\bigwedge_{\psi_i \in \Psi} ![\mathsf{tr}(\psi_i)]) \Rightarrow \mathsf{tr}(\psi)$, and try to prove it in the system R–S augmented with the appropriate rule for treating axioms on the left of '$\Rightarrow$'. The standard notion of satisfaction of such a formula is equivalent to the satisfaction of the sequence

$$\neg![\mathsf{tr}(\psi_1)], \dots, \neg![\mathsf{tr}(\psi_n)], \mathsf{tr}(\psi). \tag{1}$$

In order to reason about specifications we have to extend the R–S system by a new rule to handle the axiomatic formulae of the form $\neg![\phi]$. Notice that in (1) we do not nest axiomatic formulae, and they always occur under the negation $\neg$. Since specifications will only involve sequents over atomic formulae, we do not need the full power of universal and/or existential quantifiers. Therefore we introduce $![\_]$, resp., $\neg![\_]$ as new logical connectives which, however, are used only at the outermost level of formulae.

**Definition 12.** For a structure $M = \langle A, \alpha\rangle$ and a formula $\psi$, we define

$$\begin{array}{lll} M \models ![\psi] & \iff & A \models_* \psi \text{ (i.e., iff } \forall\alpha'.\langle A, \alpha'\rangle \models \psi\text{). Consequently} \\ M \models \neg![\psi] & \iff & M \not\models ![\psi] \iff A \not\models_* \psi \text{ (i.e., iff } \exists\alpha'.\langle A, \alpha'\rangle \models \neg\psi\text{).} \end{array}$$

$M[\alpha'/\alpha]$ denotes the structure $M$ with $\alpha$ replaced by $\alpha'$. For a formula $\phi$, we write $\phi[\overline{y}/\overline{x}]$ for $\phi$ with all the occurrences of $\overline{x}$ replaced by the respective $\overline{y}$. R–S* is obtained by augmenting R–S with the following rule:

$$\text{(AX)} \quad \frac{\Gamma', \neg![\gamma], \Gamma''}{\Gamma', \neg\gamma[\overline{y}/\overline{x}], \Gamma'', *} : \overline{x} \text{ are all variables in } \gamma \text{, and } \overline{y} \in X \text{ are arbitrary.}$$

**Remark 13.** Notice that in Definition 12 we quantify over *assignments* $\alpha'$—according to Definition 3 such an assignment may exist even if the carrier $A$ is empty, in which case all variables are assigned $\emptyset$. Note that $![\_]$ / ($\neg![\_]$ do play the role of the universal/existential closure but over assignments and not only elements of the carrier.

Consider the following special cases, with $\Gamma' = \emptyset = \Gamma''$:

(1) If $\gamma$ is $\neg(x_s \doteq x_s)$, we get

$$\text{(AX)} \quad \frac{\neg![\neg(x_s \doteq x_s)]}{\neg\neg(y_s \doteq y_s), *}\ x_s \in X.$$

Applying (IV–) to the conclusion,[3] we obtain $y_s \doteq y_s$, i.e., $\neg\mathcal{E}_s$. Thus, the formulae $\neg![\neg(x_s \doteq x_s)]$ and $\neg\neg(x_s \doteq x_s) \equiv \neg\mathcal{E}_s$, are really equivalent, moreover $\exists\alpha : x \doteq x$ is equivalent to $\exists x : x \doteq x$. (If the carrier is empty, there is not only no element but also no assignment making $x \doteq x$, since $\emptyset$ does not satisfy this equality.)

(2) If $\gamma$ is $x_s \doteq x_s$, we get

$$\text{(AX)} \quad \frac{\neg![x_s \doteq x_s]}{\mathcal{E}_s, *}\ x_s \in X,$$

where $\mathcal{E}_s$ in the conclusion corresponds to $\neg(y_s \doteq y_s)$, for some variable $y_s$ substituted for $x_s$.

[3] "–" in a reference to a rule denotes its version in the right column for the negative occurrence of the formula.

Thus, $\neg![x_s \doteq x_s]$ and $\neg(x_s \doteq x_s) \equiv \mathcal{E}_s$, are equivalent, and correspond to $\exists\alpha : \neg(x \doteq x)$ which is satisfied only by the structures with empty carrier. Note, however, that this is not equivalent to $\exists x : \neg(x \doteq x)$—this last formula is actually a contradiction.

### 3.2. Earlier treatment of empty carriers

In earlier logics for multialgebras, e.g. [12,13], one did not admit empty carrier and then $x \doteq x$ was axiomatic. The generalization with this respect amounts to having made this formula valid if and only if carrier is nonempty. The significant difference with respect to [3] is that our treatment of (non)empty carrier is essentially quantifier-free—it amounts to the treatment of the formulae $\neg(x \doteq x)$ which is carried over to the respective axioms as shown in the remark above. In [3], this required quantified formulae $\neg(\exists x.x \prec x)$.

### 3.3. The unique deduction tree

Constructing a proof for a given sequence $\Gamma = \gamma_1, \ldots, \gamma_n$ one can choose a unique, canonical deduction tree. This fact is used in the proof of completeness, but it is also of independent importance since it suggests the way of possible implementation of the logic.

The deduction tree is constructed as follows. We start with the first formula $\gamma_1$. If it is decomposable, we apply the appropriate rule, which is uniquely determined. We now check whether the obtained indecomposable formulae ("to the left'' of the active position in the obtained sequence)[4] can be used in any expansion rule and if they can, we apply the rule. If this is not possible, we apply decomposition rule—always to the "leftmost" decomposable formula in the sequence.

The following definition captures the above strategy.

**Definition 14.** An R–S* rule $\rho$ is correctly applicable to a sequence $\Gamma$ iff one of the following conditions is satisfied:
(1) $\rho$ is an R–S* rule which augments $\Gamma$ by some new formula or
(2) there is no rule with the above property that can be applied to a formula or pair of formulae that lies to the left of the (active) formula or pair of formulae to which $\rho$ is applicable.

The first point refers exclusively to the expansion rules. In the second point, $\rho$ may be a replacement rule in which case it is applied to the leftmost decomposable formula, so that no expansion rule can be applied "to the left" of it. If, in this second case, $\rho$ turns out to be an expansion rule, we see that first we have to apply the rule with one premise formula or else the rule with two premise formulae which, together, lie as far "to the left" as possible.

**Remark 15.** Since we only can use one replacement rule for a formula at any time and point 2 in definition 14 uniquely defines the expansion rule that is correctly applicable, we get that there is at most one R–S* rule that is correctly applicable to any sequence $\Gamma$ at any time.

By a deduction tree for a sequence $\Gamma$ we mean a tree with $\Gamma$ labelling the root, where the number and labelling of the children of each node originates from the application of some rule to the (sequence labelling the) node itself. Such a tree is a proof if all branches are finite and all leaves are labelled by axiomatic sequences. Alternatively, a leaf can be labelled by an indecomposable sequence to which no expansion rule is applicable. This, together with the Remark 15, leads then to the following result.[5]

**Lemma 16.** *For any sequence $\Gamma$ there is a unique decomposition tree, $DT(\Gamma)$.*

Such a unique tree will be used in the proof of completeness.

[4] Indexing formulae in a sequence as $\gamma_1, \gamma_2, \ldots, \gamma_n$, by "$\gamma_i$ lying to the left of $\gamma_j$" we mean simply that $i \leqslant j$. The "active position" is the index of the explicit formula from the premise—the one which has been processed by the rule.

[5] We assume, in general, that the set $X$ of all variables is countable. More generally, here we only need the assumption that it is well ordered, so that we can choose the first variable not present in a sequence for a fresh variable and we choose "the next variable in the ordering'' for an arbitrary variable (for each formula which is processed repeatedly, i.e., inherited as indicated by *).

### *3.4. Soundness and completeness*

As usual, soundness is easy to verify, while the proof of completeness is based on the following lemma.

**Lemma 17.** *Given a set of indecomposable formulae*, $\Gamma_{\text{ind}}$, *which is closed under all expansion rules and which does not contain an axiomatic sequence*, *there exists a counter-model* $M_{\text{C}}$ *for* $\Gamma_{\text{ind}}$, *i.e., a structure* $M_{\text{C}} = \langle A, \alpha \rangle$, *such that* $M_{\text{C}} \not\models \gamma$, *for every formula* $\gamma \in \Gamma_{\text{ind}}$.

**Proof** (*sketch*). Given such a $\Gamma_{\text{ind}}$, we define the relation $\sim$ on the set $X$ of variables by: $x \sim y \iff \neg(y \prec x) \in \Gamma_{\text{ind}}$. Closure under expansion rules implies that $\sim$ is symmetric, rule (XII), and transitive, rule (XIII).

The relation $\asymp$ is the reflexive closure of $\sim$. Again, closure under expansion rules implies that $\asymp$ is a congruence wrt. function applications: given $x \asymp y$ then if $\neg(x \prec f(z)) \in \Gamma_{\text{ind}}$, then also $\neg(y \prec f(z)) \in \Gamma_{\text{ind}}$, by rule (XIV), and if $\neg(z \prec f(x)) \in \Gamma_{\text{ind}}$, then also $\neg(z \prec f(y)) \in \Gamma_{\text{ind}}$, by rule (XV).

The counter-model $M_{\text{C}} = \langle A, \alpha \rangle$ for $\Gamma_{\text{ind}}$ is defined as follows:

(1) Carrier sets
  (a) $|A|_s = \emptyset$ iff $\neg\mathcal{E}_s \in \Gamma_{\text{ind}}$,
  (b) $|A|_s = X_s/_{\asymp}$ – otherwise.
(2) Operations: for $f : s_1 \times \cdots \times s_n \to s_{n+1}$, we define
  (a) $f([\overline{x}])^A = \emptyset$, if $|A|_{s_i} = \emptyset$ for some $1 \leqslant i \leqslant n+1$,
  (b) $f([\overline{x}])^A = \{[y] : \neg(y \prec f(\overline{x})) \in \Gamma_{\text{ind}}\}$ – otherwise.
(3) Assignment
  (a) $\alpha(x) = \emptyset$ iff $\neg\mathcal{E}_s \in \Gamma_{\text{ind}}$,
  (b) $\alpha(x) = [x]$ – otherwise.

To prove that $M_{\text{C}}$ is indeed a counter-model for $\Gamma_{\text{ind}}$, i.e., $M_{\text{C}} \not\models \gamma$, for every formula $\gamma \in \Gamma_{\text{ind}}$, one shows it for each type of the indecomposable formula (Definition 7). For instance, when $\gamma = \neg\mathcal{E}_s \in \Gamma_{\text{ind}}$, then $|A|_s = \emptyset$, and we have that $M_{\text{C}} \not\models \neg\mathcal{E}_s$. When $\gamma = \mathcal{E}_s \in \Gamma_{\text{ind}}$ then, since $\Gamma_{\text{ind}}$ is nonaxiomatic, it means that $\neg\mathcal{E}_s \notin \Gamma_{\text{ind}}$, hence: $|A|_s = X/_{\asymp} \neq \emptyset$, so $M_{\text{C}} \not\models \mathcal{E}_s$.

Completing the case analysis gives the result.

Using this lemma, we obtain the main completeness theorem.

**Theorem 18.** *The R–S* proof system is complete: for every sequence* $\Gamma$ (*possibly*, *of the form* (1)): $\models \Gamma \iff \vdash \Gamma$.

**Proof** (*sketch*). We show that if $\not\vdash \Gamma$, then there exists a counter-model $M_C \not\models \Gamma$. Let $DT(\Gamma)$ be the unique decomposition tree for $\Gamma$ as described in 3.3. There are two situations when $DT(\Gamma)$ is not a proof:

I. Some leaves are labelled by nonaxiomatic sequences—then such leaves have labels containing only indecomposable formulae. Take the label of one of such leaves as $\Gamma_{\text{ind}}$—by definition of $DT(\Gamma)$, $\Gamma_{\text{ind}}$ is closed under all expansion rules. Since $M \models \Gamma$ implies $M \models \Gamma_{\text{ind}}$, Lemma 17, giving a counter-model $M_C \not\models \Gamma_{\text{ind}}$, implies that $M_C \not\models \Gamma$.
II. The tree is infinite, which implies (by the König lemma) that there exists an infinite branch. Select such an infinite branch $B$ from $DT(\Gamma)$, and collect all indecomposable formulae occurring along it into the set $\Gamma_{\text{ind}}$. Since $B$ is infinite, $\Gamma_{\text{ind}}$ does not contain any axiomatic sequence. Also $\Gamma_{\text{ind}}$ is closed under all expansion rules (for any (pair of) indecomposable formula(e) will be selected, along an infinite branch, for a correct application of an appropriate (if any) expansion rule). Thus $\Gamma_{\text{ind}}$ satisfies the conditions of Lemma 17, so let $M_{\text{C}}$ be the counter-model as it was defined in the proof of this lemma. We show that $M_{\text{C}}$ is a counter-model for all the formulae occurring in the labels of the vertices of $B$, and since the start vertex of $B$ is the root of $DT(\Gamma)$ labelled with $\Gamma$, we have that $M_{\text{C}}$ is a counter-model for $\Gamma$.

The proof goes by induction on the rank of a formula $\gamma$, $ord(\gamma)$, which is defined so that the applications of rules never increase the rank of the formulae in the sequence and, eventually, decrease it. First, for all indecomposable $\gamma$ we let $ord(\gamma) = 0$. Then, for instance, $ord(x \prec t) = ord(\neg(x \prec t)) = 1$ (where $t \notin X$, $x \in X$), $ord(t \prec t') = ord(\neg(t \prec t')) = 2$ (where $t \notin X$), $ord(\neg![\gamma']) = ord(\neg\gamma') + 1$, etc.

Now, if $M_{\text{C}} \models \Gamma$, then the set $\Gamma_{\text{sat}}$ of all formulae $\gamma'$, appearing in one of the vertices of $B$ and such that $M_{\text{C}} \models \gamma'$, is nonempty, since $\Gamma \cap \Gamma_{\text{sat}} \neq \emptyset$. Let $\gamma_i \in \Gamma_{\text{sat}}$ be such that $ord(\gamma_i) \leqslant ord(\gamma')$, for every $\gamma' \in \Gamma_{\text{sat}}$. One shows, by induction on the rank of $\gamma_i$, that it must be indecomposable. But then $\gamma_i \in \Gamma_{\text{ind}}$ and so, by Lemma 17, $M_C \not\models \gamma_i$, contradicting the assumption. □

**Corollary 19.** *A sequence $\Gamma$ has a proof in the R–S* system iff $DT(\Gamma)$ is a proof.*

**Proof.** The 'if' part is trivial and the 'only if' part follows from the proof of the above theorem. If $DT(\Gamma)$ is not a proof, then we have a counter-model for $\Gamma$. Since R–S* is sound, we conclude that $\Gamma$ is not provable. □

Combining it with soundness, and writing $\Psi \vdash \psi$ for $\vdash \neg ![\psi_1], \ldots, \neg ![\psi_n], \psi$ gives:

**Corollary 20.** *For any formula $\phi$ and finite set of formulae $\Psi = \{\psi_1, \ldots, \psi_n\} : \Psi \vdash \phi \iff \Psi \models_* \phi$.*

We have thus obtained the sound and complete system for proving consequences of specifications. As remarked, the system R–S*, with the unique proof strategy described in Section 3.3, is well suited for implementation. It is, however, less convenient for doing proofs by hand. In the following section we make the last step and design a Gentzen system which provides much simpler means for performing proofs by hand—it works directly with sequents and does not require any translation of sequents into formulae.

## 4. Gentzen calculus

We describe first a trivial translation of the R–S* system into a Gentzen system GS*. The final Gentzen system GS is then obtained by some further simplifications and a specific transformation of the rules (AX).

**Definition 21.** A formula $\gamma$ is negative if it has the form $\neg\gamma'$. For any sequence $\Gamma$ we define: $\Gamma^+ = \{\gamma \in \mathcal{F}_\Sigma : \gamma \text{ is nonnegative and } \gamma \in \Gamma\}$ and $\Gamma^- = \{\gamma \in \mathcal{F}_\Sigma : \neg\gamma \in \Gamma\}$.

Lemma 9 can be now rephrased in the following way:

**Lemma 22.** *A sequence $\Gamma$ is valid iff the sequent $\Gamma^- \Rightarrow \Gamma^+$ is valid.*

The R–S* rules, inverted and reformulated according to this corollary, give a sound Gentzen system. For instance, the rule

$$\text{(VII)} \quad \frac{\Gamma', t \prec t', \Gamma''}{\Gamma', \neg(x \prec t), x \prec t', \Gamma''}, \quad \text{where } t \notin X, \text{ and } x \in X \text{ is fresh}$$

gives rise to the Gentzen rule:

$$\frac{\Gamma, x \prec t \Rightarrow x \prec t', \Delta}{\Gamma \Rightarrow t \prec t', \Delta}, \quad \text{where } t \notin X, \text{ and } x \in X \text{ is fresh.}$$

To obtain completeness of the translated system one has to add one swapping rule (the other one emerges from (IV)): $\frac{\Gamma, \gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg\gamma}$ —the resulting system is denoted GS*. The GS system we are aiming at has some more fundamental differences from R–S*:

- The rules in GS are not only applicable "bottom up'' but,
- they are not invertible—generally, they are sound only "top down'',
- Sequents are pairs of *sets* of formulae, where ordering is ignored,
- One can derive consequences also of infinite sets of axioms.

Now, assuming that all our sequents are as indicated in the specifications, i.e., contain only atomic formulae, we perform the crucial transformation to obtain a "pure" sequent calculus for specifications, i.e., one operating only on sequents of atomic formulae and allowing to derive such sequents from specifications without any coding and translation. The transformation concerns the rules (AX).

For the sake of example, assume a specification containing only one sequent, $\Psi = \{\gamma \Rightarrow \delta\}$. To derive from it $\Gamma \Rightarrow \Delta$ in GS*, one would try to prove $![\neg\gamma \vee \delta], \Gamma \Rightarrow \Delta$ which would be processed with the following application of the rule (AX), where $\overline{y}$'s match the respective variables $\overline{x}$ from $\gamma \Rightarrow \delta$:

$$\frac{(\neg\gamma \vee \delta)[\overline{y}/\overline{x}], \Gamma \Rightarrow \Delta}{![\neg\gamma \vee \delta], \Gamma \Rightarrow \Delta}, \quad \overline{y} \text{ arbitrary.} \tag{2}$$

Applying now the rules for disjunction (V–) and negation (IV–) in the antecedent, we will end up with the assumptions as indicated below:

$$\frac{\delta[\overline{y}/\overline{x}], \Gamma \Rightarrow \Delta \mid \Gamma \Rightarrow \Delta, \gamma[\overline{y}/\overline{x}]}{![\neg\gamma \vee \delta], \Gamma \Rightarrow \Delta}, \quad \overline{y} \text{ arbitrary.} \tag{3}$$

All the assumptions are now sequents and this illustrates the idea of the final step.

Since our sequents have only atomic formulae, we can now remove the rules for all the connectives: negation (IV), disjunction (V) and conjunction (VI). Moreover, we remove the axiom rule (AX) and introduce instead the rules of *specific cut*, [10], for each nonlogical axiom $\gamma_1, \ldots, \gamma_n \Rightarrow \delta_1, \ldots, \delta_m \in \Psi$:

$$\text{(SPC)} \quad \frac{\Gamma \Rightarrow \Delta, \gamma'_1 \mid \ldots \mid \Gamma \Rightarrow \Delta, \gamma'_n \mid \Gamma, \delta'_1 \Rightarrow \Delta \mid \ldots \mid \Gamma, \delta'_m \Rightarrow \Delta}{\Gamma \Rightarrow \Delta},$$

where the primed versions denote a uniform, arbitrary renaming of variables occurring in the involved axiom $\gamma_1, \ldots, \gamma_n \Rightarrow \delta_1, \ldots, \delta_m \in \Psi$.

As argued in [10], the specific cut rules are significantly more manageable than the general cut. In fact, the "undecidability" of such rules (applied bottom-up) is essentially of the same kind as that of the axiom rules (AX) and concerns only the choice of the appropriate variable names.

The rules of the resulting system GS are given below. We cannot claim the equivalence of GS* and GS, since the latter does not allow any non-logical axioms. However, taking into account the restrictions on such formulae we have put in GS* (only $\neg![\ldots]$ occurring only at the outermost level, with the exception of one formula, corresponding to the sequent we are proving), the above remarks make it obvious that, for any sequents $\psi_1, \ldots, \psi_n, \psi$ over atomic formulae, we have:

$$\vdash_{GS^*} ![\mathsf{tr}(\psi_1)], \ldots, ![\mathsf{tr}(\psi_n)] \Rightarrow \mathsf{tr}(\psi) \iff \{\psi_1, \ldots, \psi_n\} \vdash_{GS} \psi,$$

Indeed, if there is a proof in GS* involving an application of (AX), as in (2), then, moving "bottom-up'', it must split the tree into branches for separate disjuncts (of each $\mathsf{tr}(\psi_i)[\overline{y}/\overline{x}]$) before processing the involved disjuncts themselves. Hence it must pass through nodes as given in the assumptions of (3). Except for the superficial differences of syntax, the rule (SPC) mimics exactly transition to such nodes. On the other hand, the rule is obviously sound (with the interpretation of $\{\psi_1 \ldots \psi_n\} \vdash \psi$ as $\{\psi_1 \ldots \psi_n\} \models_* \psi$), and hence it is admissible in GS*.

**Theorem 23.** *The system* GS *given below is sound and strongly complete*, *i.e.*, *for a finite specification* $\Psi$ *and sequent* $\psi$: $\Psi \models_* \psi \iff \Psi \vdash_{GS} \psi$.

We have thus obtained the calculus GS for deriving consequences of specifications, which does not require any transformation of the involved sequents—its axioms and rules are given below. Notice also that, although infinite specifications require infinitely many (SPC) rules, these are still possible to handle along the same lines as the finite specifications, unlike the infinite formulae which would result from the application of the R–S* strategy. [6]

*Axioms*:

(I) $\Gamma \Rightarrow x \prec x, \Delta \ : x \in X$, (II) $\Gamma, \gamma \Rightarrow \gamma, \Delta$ and (III) $\Gamma, \mathcal{E} \Rightarrow t_s \prec t'_s, \Delta$.

[6] We do not claim strong completeness also for infinite theories, but only the possibility of using GS for deriving consequences of such theories.

*Replacement rules*:

(VII) $$\frac{\Gamma, x \prec t \Rightarrow \Delta, x \prec t'}{\Gamma \Rightarrow \Delta, t \prec t'}, \qquad \frac{\Gamma \Rightarrow \Delta, x \prec t \mid \Gamma, x \prec t' \Rightarrow \Delta}{\Gamma, t \prec t' \Rightarrow \Delta},$$

$t \notin X$, and $x \in X$ is fresh, $\qquad$ $t \notin X$ and $x \in X$ arbitrary,

(VIII) $$\frac{\Gamma \Rightarrow \Delta, y \prec t \mid \Gamma \Rightarrow \Delta, x \prec f(\ldots, y, \ldots)}{\Gamma \Rightarrow \Delta, x \prec f(\ldots, t, \ldots)}, \qquad \frac{\Gamma, y \prec t, x \prec f(\ldots, y, \ldots) \Rightarrow \Delta}{\Gamma, x \prec f(\ldots, t, \ldots) \Rightarrow \Delta},$$

where $y \in X$ arbitrary and $t \notin X$, $\qquad$ where $y \in X$ is fresh and $t \notin X$,

(IX) $$\frac{\Gamma \Rightarrow \Delta, t \doteq x \mid \Gamma \Rightarrow \Delta, t' \doteq x}{\Gamma \Rightarrow \Delta, t \doteq t'}, \qquad \frac{\Gamma, t_s \doteq x_s, t'_s \doteq x_s \Rightarrow \Delta}{\Gamma, t_s \doteq t'_s \Rightarrow \Delta},$$

$t, t' \notin X$ and $x \in X$ arbitrary, $\qquad$ $t_s, t'_s \notin X$ and $x_s \in X$ is fresh.,

(X) $$\frac{\Gamma \Rightarrow \Delta, t_s \prec x_s \mid \Gamma \Rightarrow \Delta, x_s \prec t_s \mid \Gamma, \mathcal{E}_s \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, t_s \doteq x_s}, \qquad \frac{\Gamma, t_s \prec x_s, x_s \prec t_s \Rightarrow \Delta, \mathcal{E}_s}{\Gamma, t_s \doteq x_s \Rightarrow \Delta},$$

where $x_s \in X$ and $t_s \neq x_s$, $\qquad$ where $x_s \in X$ and $t_s \neq x_s$,

(XI) $$\frac{\Gamma \Rightarrow \Delta, t_s \prec x_s \mid \Gamma \Rightarrow \Delta, x_s \prec t_s \mid \Gamma, \mathcal{E}_s \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, x_s \doteq t_s}, \qquad \frac{\Gamma, t_s \prec x_s, x_s \prec t_s \Rightarrow \Delta, \mathcal{E}_s}{\Gamma, x_s \doteq t_s \Rightarrow \Delta},$$

where $x_s \in X$ and $t_s \neq x_s$, $\qquad$ where $x_s \in X$ and $t_s \neq x_s$.

*Specific cut rules* (*for each axiom* $\gamma_1, \ldots, \gamma_n \Rightarrow \delta_1, \ldots, \delta_m \in \Psi$):

(SPC) $$\frac{\Gamma \Rightarrow \Delta, \gamma'_1 \mid \ldots \mid \Gamma \Rightarrow \Delta, \gamma'_n \mid \Gamma, \delta'_1 \Rightarrow \Delta \mid \ldots \mid \Gamma, \delta'_m \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}$$

with an arbitrary *uniform* renaming $'$ of variables across all $\gamma_i, \delta_j$.

*Expansion rules* ((XIV), (XV) *sound for arbitrary* $z$):

(XII) $$\frac{\Gamma, x \prec y \Rightarrow \Delta}{\Gamma, y \prec x \Rightarrow \Delta},$$ (XIII) $$\frac{\Gamma, y \prec z \Rightarrow \Delta}{\Gamma, y \prec x, x \prec z \Rightarrow \Delta},$$

(XIV) $$\frac{\Gamma, y \prec f(\overline{x}) \Rightarrow \Delta}{\Gamma, y \prec z, z \prec f(\overline{x}) \Rightarrow \Delta},$$ (XV) $$\frac{\Gamma, x \prec f(\ldots, y, \ldots) \Rightarrow \Delta}{\Gamma, y \prec z, x \prec f(\ldots, z, \ldots) \Rightarrow \Delta},$$

(XVI) $$\frac{\Gamma, \mathcal{E}_s, x_{s'} \prec f(\ldots, y_s, \ldots), \mathcal{E}_{s'} \Rightarrow \Delta}{\Gamma, \mathcal{E}_s, x_{s'} \prec f(\ldots, y_s, \ldots) \Rightarrow \Delta}.$$

## 5. Conclusions

We have applied the technique of Rasiowa–Sikorski (R–S) from [11] for designing sound and complete cut-free logics for deriving multialgebraic tautologies. We have augmented the system with the rules for deriving consequences of (finite) theories. For the more specific goal of deriving consequences of multialgebraic specifications written using only sequents of atomic formulae, we have given a more user-friendly system GS without general cut, in which axioms give rise to the specific cut rules, inspired by Pliuškevičienė [10]. Both techniques are generic and can be applied in many situations other than multialgebraic reasoning studied in this paper.

As compared to the most closely related work which also used the R–S technique in the context of multialgebras, [3], the main difference is the presence of the new predicate, $\doteq$, which was not included in the language of [3]. We have argued why this predicate is relevant and useful, especially, when specifying nondeterministic data types. We have also shown how (non)empty carriers can be treated using this predicate instead of quantifiers needed in [3]. Furthermore, the logic from [3] allows one to derive only tautologies but not logical consequences of sets of given, nonlogical axioms. We have elaborated the possibility (only implicit in [2,3]) of extending logic for such purpose, by providing the required translation schema. Then, we have shown how this translation schema (as well as rules for connectives and axioms), needed to handle nonlogical axioms in the R–S$^*$ system, can be removed and replaced by the specific cut rules. The resulting system can be used directly, without any intermediary transformations, for deriving consequences from specifications. It is essentially quantifier-free—following the tradition of algebraic specifications, it handles only implicitly the universal closure of axioms.

## Acknowledgments

Both authors gratefully acknowledges the financial support received from the Norwegian Research Council, NFR, under the MoSIS project.

## References

[1] A. Avron, B. Konikowska, Decomposition systems for gödel-dummett logics, Studia Logica, special issue: Analytic Proof Techniques, 2000.
[2] M. Białasik, B. Konikowska, A logic for non-deterministic specifications, in: E. Orłowska (Ed.), Logic at Work Essays Dedicated to the Memory of H. Rasiowa, Springer, Berlin, 1998, pp. 286–311.
[3] M. Białasik, B. Konikowska, Reasoning with first-order nondeterministic specifications, Acta Inform. 36 (1999) 357–403.
[4] J.A. Goguen, R.M. Burstal, Institutions: abstract model theory for specification and programming, J. ACM 39 (1992) 95–146.
[5] H. Hussmann, Nondeterminism in Algebraic Specifications and Algebraic Programs, Birkhäuser, Basel, 1993.
[6] B. Konikowska, Rasiowa-Sikorski deduction systems: a handy tool for computer science logic, in: J. Fiadeiro (Ed.), Recent Trends in Algebraic Specification Techniques, Lecture Notes in Computer Science, vol. 1589, Springer, Berlin, 1999.
[7] B. Konikowska, Rasiowa-Sikorski deduction systems in computer science applications, Theoret. Comput. Sci. 286 (2) (2002) 323–366.
[8] Y. Lamo, The institution of multialgebras—a general framework for algebraic software development, Ph.D. Thesis, Department of Informatics, University of Bergen, 2002.
[9] Y. Lamo, M. Walicki, Modeling partiality by nondeterminism, in: N. Callaos, J.M. Pineda, M. Sanchez (Eds.), Proc. SCI/ISAS 2001, vol. I, Orlando, FL, 2001.
[10] A. Pliuškievičienė, Specialization of the use of axioms for deduction search in axiomatic theories with equality, J. Soviet Math. 1 (1973).
[11] H. Rasiowa, R. Sikorski, The Mathematics of Metamathematics, PWN (Polish Scientific Publishers), 1963.
[12] M. Walicki, S. Meldal, A complete calculus for the multialgebraic and functional semantics of nondeterminism, ACM TOPLAS 17 (2) (1995).
[13] M. Walicki, S. Meldal, Multialgebras, power algebras and complete calculi of identities and inclusions, in: Recent Trends in Algebraic Specification Techniques, Lecture Notes in Computer Science, vol. 906, Springer, Berlin, 1995.
[14] M. Walicki, S. Meldal, Algebraic approaches to nondeterminism—an overview, ACM Comput. Surveys 29 (1997).