Quantifier-free logic for multialgebraic theories

Yngve Lamo[1]

Faculty of Engineering, Bergen University College, 5020 Bergen, Norway,
yla@hib.no

Michał Walicki[1]

Department of Informatics, University of Bergen, 5020 Bergen, Norway,
michal@ii.uib.no

Abstract

We develop a new quantifier-free logic for deriving consequences of multialgebraic theories. Multilagebras 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 complete Rasiowa-Sikorski logic for proving multilagebraic 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 complete Gentzen system for proving directly consequences of specifications. The improvements over earlier logics for multialgebras concern mainly the ability to handle empty carriers (as well as empty result-sets) without the use of quantifiers, and to derive consequences of (potentially infinite) theories without the use of general cut.

Introduction

The institution [4] of multialgebras, $\mathcal{MA}$ [9], provides a powerful algebraic framework for specification – primarily, but not exclusively, of nondeterministic behavior [6,9,14,15]. 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 [10]. The logic has two atoms: set inclusion $t \prec t'$ holds iff the interpretation of t is included

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

This is a preliminary version. The final version will be published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

in the interpretation of t', 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 from [12], 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 [2]. However, their language does not include the element equality $\doteq$. This predicate can't be expressed in their 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. E.g., $\exists x : x \prec x$ expresses non-emptiness of the carrier, which in our language, can be expressed by the quantifier-free formula $x \doteq x$ (with only implicit universal quantification over possible assignments). Finally, and most significantly, the language from [2], unless extended to full first-order, is not expressive enough to state non-emptiness 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 aspects. 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, 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.

Having introduced the basic notions from multialgebras in Section 1, we design a sound and complete R-S system in Section 2. Following [12] (and also [1,2,7,8]) 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 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 [11]. 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 [2]. Details and proofs can be found in [9].

1 Multialgebras

Multialgebras are presented using the standard algebraic signatures: $\Sigma = (S, \Omega)$, where S is a set of sort names and Ω a set of operation names with profiles sorted over S. Terms over a signature Σ 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: $t_s \prec t'_s$ and $t_s \doteq t'_s$, where $t_s, t'_s \in \mathcal{T}_{\Sigma,X}$.

Definition 1.1 *A Σ-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 $f : s_1 \times \ldots \times s_n \to s$ we have: $f^A : s_1^A \times \ldots \times s_n^A \to \mathcal{P}(s^A)$, 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. An operation that is neither partial nor nondeterministic is a function.

Definition 1.2 *Given a multialgebra A, an assignment α is a function $\alpha : X \to |A| \uplus \{\emptyset\}$ where $\alpha(x_s) = \emptyset \iff s^A = \emptyset$.* [2]

A Σ structure $M = \langle A, \alpha\rangle$ is a Σ multialgebra A with an assignment α.

So an assignment assigns an element – not a *set* of elements! – to each variable of a nonempty sort. 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 1.3 *Satisfaction in a structure $M = \langle A, \alpha\rangle$ is defined by:*

(i) $M \models t \prec t' \iff t^M \subseteq t'^M$

(ii) $M \models t \doteq t' \iff t^M = \{e\} = t'^M$, *for some* $e \in |A|$ [3]

(iii) $M \models \neg\gamma \iff M \not\models \gamma$, *and* $M \models \gamma \vee \phi \iff M \models \gamma$ *or* $M \models \phi$.

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

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

According to point ii of def. 1.3, an equality may hold only if the carrier is non-empty. Given a structure $M = \langle A, \alpha\rangle$, we have that:

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

2 The R-S calculus

We present a quantifier free Rasiowa-Sikorski deduction system, R-S, for multialgebras. It illustrates a powerful way of designing deduction systems based

[2] Alternatively, an assignment could be a partial function, with domain being the variables of nonempty sorts. Our formulation gives that any non-ground term with variables from empty sort will be empty, as multialgebraic operations are strict on the empty set.

[3] We do not distinguish one-element set $\{e\}$ and the element e.

on the semantical properties of the atoms, originally introduced in [12]. Our presentation is an adaptation and extension of a similar logic described in [2].

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 without repetitions.

Definition 2.1 *A structure $M = \langle A, \alpha \rangle$ satisfies a sequence $\Gamma = \gamma_1, \ldots, \gamma_n$, 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 Γ_i are sequences:

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

Both sides of ” $\mid$ ” have to hold for making an expression involving $\mid$ true, hence it should be viewed as a meta-conjunction. The rules are invertible.

Definition 2.2 *An (R-S) rule is sound when, for any structure M, M satisfies the premise iff it satisfies the conclusion.*

Particular sequences are singled out as *axiomatic*, in our case, sequences containing a formula or a subsequence of the form (ignoring the order):

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

In addition to axiomatic sequences, one also identifies the *indecomposable* sequences – in our case, these are given by the following definition.

Definition 2.3 *A Σ 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 $x, y \in X$*
- *$x \prec f(x_1, \ldots, x_n)$ or $\neg(x \prec f(x_1, \ldots, x_n))$, where $x, x_i \in X$ and $f \in \Omega$*

A sequence Γ is indecomposable iff every formula in Γ is indecomposable.

No rule can modify an indecomposable formulae – if such a formula appears during the proof, it will stay unchanged for the rest of the proof.

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 (i.e., expressions involving only variables or function application to variables). 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, and one for its negative version.

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

The sign “*” in the conclusion of a rule indicates repetition of the active formula from the premise.

2.1 The R-S proof system

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

(I) $x \prec x \ :x \in X$ (II) $\gamma, \neg\gamma \ :\gamma \in \mathcal{F}_{\Sigma,X} \cup \{\mathcal{E}\}$ (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 $\qquad$ $t \notin X$ 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 $t \notin X$ $\qquad$ $y \in X$ 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 $\qquad$ $t, t' \notin X$ 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 $t_s \neq x_s$ $\qquad$ $x_s \in X$ 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 $t_s \neq x_s$ $\qquad$ $x_s \in X$ 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) $$\frac{\Gamma', \neg(y \prec x), \Gamma'', \neg(x \prec f(\overline{z})), \Gamma'''}{\Gamma', \neg(y \prec x), \Gamma'', \neg(x \prec f(\overline{z})), \neg(y \prec f(\overline{z})), \Gamma'''}$$

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

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

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

2.2 Unique deduction tree and completeness

Given a 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.

We start with the first formula γ_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) 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" formula in the sequence.

By a deduction tree for a sequence Γ we mean a tree with Γ 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 leaves are labelled by axiomatic sequences. The above idea leads then to the following result.

Lemma 2.4 *For any sequence Γ there is a unique decomposition tree, $DT(\Gamma)$.*

Theorem 2.5 *The R-S system is sound and complete: $\models \Gamma \iff \vdash \Gamma$.*

The proof of completeness utilizes lemma 2.4. As a consequence, we also obtain that Γ has a proof in the R-S system iff $DT(\Gamma)$ is a proof.

3 Specifications and system R-S*

The system R-S can be used to derive only tautologies – valid sequences. But we are really interested in proving logical consequences of specifications. A specification $\mathcal{SP}$ is pair (Σ, Ψ), where Σ is a signature (kept implicit in what follows) and Ψ 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 complete Gentzen system.

A Σ sequent is a pair, written $\Gamma \to \Delta$, of finite sets of formulae from $\mathcal{F}_{\Sigma,X}$. The notation means, implicitly, $\gamma_1, \ldots, \gamma_n \to \delta_1, \ldots, \delta_m$. As a matter of fact, following earlier works, e.g. [6,14,9], our specifications involve only sequents of *atomic* formulae (i.e., each γ_i, δ_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 3.1 *A sequent* $\Gamma \to \Delta$ *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* $M \models \delta_j$*.*

Lemma 3.2 *A sequent* $\Gamma \to \Delta$ *is valid iff the sequence* $\neg\Gamma, \Delta$ *is valid.*

The latter notation stands for the sequence $\neg\gamma_1, \ldots, \neg\gamma_n, \delta_1, \ldots, \delta_m$.

Definition 3.3 *The function* tr *translates sequents to formulae in* $\mathcal{F}_{\Sigma,X}$*:*

- $\mathsf{tr}(\gamma_1, \ldots, \gamma_n \to \delta_1, \ldots, \delta_m) \equiv \neg\gamma_1 \vee \ldots \vee \neg\gamma_n \vee \delta_1 \vee \ldots \vee \delta_m$.
- *for* $\Psi = \{\psi_1, ..., \psi_n\} : \mathsf{tr}(\Psi) = \{\mathsf{tr}(\psi_1), ..., \mathsf{tr}(\psi_n)\}$

With the above notation, lemma 3.2 can be stated as: for any structure M and sequent $\psi : M \models \psi \iff M \models \mathsf{tr}(\psi)$

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

Definition 3.4 *Given a specification* $\mathcal{SP} = (\Sigma, \Psi)$*, a* Σ*-multialgebra* A*, and a sequent* ψ*, we define the satisfaction relation* $\models_*$*:*

(i) $A \models_* \psi \iff \forall\alpha.\langle A, \alpha \rangle \models \mathsf{tr}(\psi)$

(ii) $A \models_* \Psi \iff \forall\psi_i \in \Psi. A \models_* \mathsf{tr}(\psi_i)$

(iii) $\Psi \models_* \psi \iff \forall A.(A \models_* \Psi \Rightarrow A \models_* \mathsf{tr}(\psi))$.

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

*3.1 The system R-S**

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

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

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 3.5 *For a structure $M = \langle A, \alpha \rangle$ and a formula ψ, we define:*

$$M \models \,![\psi] \iff A \models_* \psi \text{ (i.e., iff } \forall \alpha'.\langle A, \alpha' \rangle \models \psi\text{). Consequently:}$$

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

$M[\alpha'/\alpha]$ denotes the structure M with α replaced by α'. For a formula ϕ, we write $\phi[\overline{y}/\overline{x}]$ for ϕ 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 3.6 *Notice that in def. 3.5 we quantify over* assignments α' *– according to def. 1.2 such an assignment may exist even if the carrier A is empty, in which case all variables are assigned $\emptyset$. $![\_]/\neg![\_]$ 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 $\Gamma' = \emptyset = \Gamma''$:

(i) *If γ is $\neg(x_s \doteq x_s)$, we get:*

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

Applying (IV–) to the conclusion, 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, i.e. $\exists\alpha : x \doteq x$ 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.)

(ii) *If γ is $x_s \doteq x_s$, we get:*

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

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

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.

In earlier logics of ours, e.g. [13,14], we 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 non-empty. The significant difference with respect to [2] 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 [2], this required quantified formulae $(\neg)\exists x.x \prec x$.

Lemma 3.7 *The R-S* proof system is sound and complete: for any sequence Γ (possibly, of the form (1)), $\models \Gamma \iff \vdash \Gamma$.*

Writing $\Psi \vdash \psi$ for $\vdash \neg![\psi_1], \ldots, \neg![\psi_n], \psi$ and $\Psi \models \psi$ for $\models \neg![\psi_1], \ldots, \neg![\psi_n], \psi$, the above lemma gives us:

Theorem 3.8 *For any formula ϕ 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 2.2, 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 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 will first describe a trivial translation of the R-S* system into a Gentzen system. The final Gentzen system GS, given in Section 4.1, will be obtained by some further simplifications.

Definition 4.1 *A formula γ is negative if it has the form $\neg\gamma'$. For any sequence Γ we define: $\Gamma^+ = \{\gamma \in \mathcal{F}_\Sigma : \gamma$ is non-negative and $\gamma \in \Gamma\}$ and $\Gamma^- = \{\gamma \in \mathcal{F}_\Sigma : \neg\gamma \in \Gamma\}$*

We can now rephrase lemma 3.2 in the following way:

Corollary 4.2 *A sequence Γ is valid iff the sequent $\Gamma^- \to \Gamma^+$ is valid.*

The GS system has some fundamental differences from the R-S system:

- The rules in GS are applied "bottom up".
- The GS rules aren't 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.

We use corollary 4.2 on the different types of R-S rules and get a sound Gentzen system. To obtain completeness of the translated system one has to add one swapping rule: $\dfrac{\Gamma, \gamma \to \Delta}{\Gamma \to \Delta, \neg\gamma}$ – this system is called GS".

4.1 The final Gentzen system GS

Assuming that all our sequents are as indicated in the specifications, i.e., contain only atomic formulae, and observing that function $\mathtt{tr}$ (def. 3.3) introduces only disjunctions, the soundness and completeness holds also when we remove from GS" both rules (VI). We now perform a final 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 translation nor axiom rules.

For the sake of example, let our specification contain only one sequent, $\Psi = \{\gamma \to \delta\}$. To derive from it $\Gamma \to \Delta$, we would try to prove $![\neg\gamma \vee \delta], \Gamma \to \Delta$ which, applying the rule (AX), amounts to application i. on the left, where $\overline{y}$'s match the respective variables $\overline{x}$ from $\gamma \to \delta$:

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

Applying the rules for disjunction (V–) and negation (IV–) in the antecedent, we will end up with the assumptions as indicated in the ii. above. All the assumptions are now sequents and this illustrates the idea of the final step. We can remove the rules for the connectives: (IV), (V) and (VI). Moreover, we remove also the axiom rule (AX) and introduce instead the rules of *specific cut*, [11], for each non-logical axiom $\gamma_1...\gamma_n \to \delta_1...\delta_m \in \Psi$:

$$\text{(SPC)}\ \frac{\Gamma \to \Delta, \gamma_1' \mid ... \mid \Gamma \to \Delta, \gamma_n' \mid \Gamma, \delta_1' \to \Delta \mid ... \mid \Gamma, \delta_m' \to \Delta}{\Gamma \to \Delta}$$

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

As argued in [11], 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 can not claim the equivalence of GS" and GS, since the latter does not allow any formulae with 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

$$\vdash_{GS''} ![\mathsf{tr}(\psi_1)], \ldots, ![\mathsf{tr}(\psi_n)] \to \mathsf{tr}(\psi) \iff \{\psi_1, \ldots, \psi_n\} \vdash_{GS} \psi,$$

for any sequents $\psi_1, \ldots, \psi_n, \psi$ over atomic formulae. Indeed, if there is a proof in GS" involving an application of (AX), as in (2).i, 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 (2).ii. 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 $\models ![\mathsf{tr}(\psi_1)], \ldots, ![\mathsf{tr}(\psi_n)] \to \mathsf{tr}(\psi)$), and hence it is admissible in GS".

We thus obtain the calculus GS for deriving consequences of specifications, which does not require any transformation of the involved sequents, and the following theorem follows.

Theorem 4.3 *The system GS given below is sound and complete, i.e., for any specification Ψ and sequent ψ: $\Psi \models_* \psi \iff \Psi \vdash_{GS} \psi$.*

Axioms

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

Replacement rules

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

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

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

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

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

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

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

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

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

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 $\gamma_1, \ldots, \gamma_n \to \delta_1, \ldots, \delta_m \in \Psi$)

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

with arbitrary uniform renaming $'$ of variables in all γ_i, δ_j

Expansion rules ((XIV), (XV) sound for arbitrary z)

(XIV) $$\frac{\Gamma, y \prec f(\overline{x}) \to \Delta}{\Gamma, y \prec z, z \prec f(\overline{x}) \to \Delta}$$ (XII) $$\frac{\Gamma, x \prec y \to \Delta}{\Gamma, y \prec x \to \Delta}$$

(XV) $$\frac{\Gamma, x \prec f(\ldots, y, \ldots) \to \Delta}{\Gamma, y \prec z, x \prec f(\ldots, z, \ldots) \to \Delta}$$ (XIII) $$\frac{\Gamma, y \prec z \to \Delta}{\Gamma, y \prec x, x \prec z \to \Delta}$$

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

5 Conclusions

We have applied the technique of Rasiowa-Sikorski [12] for designing sound and complete cut-free logics for reasoning about multialgebras. We hope that this paper, may draw more attention to this elegant and powerful technique, and facilitate its broader applications. More details on and applications of this technique can be found in [1,8,7].

As compared to the most closely related work which also used this technique, [2], the main difference is the presence of the new predicate, $\doteq$, which was not included in the language of [2]. 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 [2]. Furthermore, the logic from [2] allows one to derive only tautologies but not logical consequences of sets of given, non-logical axioms. We have elaborated the possibility (only implicit in [2]) 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 non-logical axioms in the R-S* system (and in [2]), can be removed and replaced by the specific cut rules, inspired by [11]. 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.

The unique decomposition tree which provides a proof strategy and has been identified for the introduced logics R-S and R-S*, following [12], is a natural candidate for a possible implementation and we expect that such an implementation will become available in not too far future.

References

[1] Arnon Avron and Beata Konikowska. Decomposition systems for gödel-dummett logics. *Studia Logica, special issue: Analytic Proof Techniques*, 2000.

[2] Marcin Białasik and Beata Konikowska. Reasoning with first-order nondeterministic specifications. *Acta Informatica*, 36:357–403, 1999.

[3] Joseph Goguen, James Thatcher, and Eric Wagner. An initial algebra approach to the specification, correctness, and implementation of abstract data types. In Raymond Yeh, editor, *Current Trends in Programming Methodology*, volume 4, pages 80–149. Prentice-Hall, 1978.

[4] Joseph A. Goguen and R. M. Burstal. Institutions: Abstract model theory for specification and programming. *Journal of the ACM*, 39:95–146, 1992.

[5] J.V. Guttag. *The specification and Application to Programming of Abstract datatypes*. PhD thesis, University of Toronto, 1975.

[6] Heinrich Hussmann. *Nondeterminism in Algebraic Specifications and Algebraic Programs*. Birkhäuser, 1993.

[7] Beata Konikowska. Rasiowa-sikorski deduction systems: a handy tool for computer science logic. In J. Fiadeiro, editor, *Recent Trends in Algebraic Specification Techniques*, volume 1589 of *LNCS*. Springer, 1999.

[8] Beata Konikowska. Rasiowa-sikorski deduction systems in computer science applications. Technical Report 916, IPI PAN, Warsaw, 2000. [to appear in Theoretical Computer Science, special issue: Algebraic Development Techniques].

[9] Yngve Lamo. *The institution of multialgebras – a general framework for algebraic software development*. PhD thesis, Department of Informatics, University of Bergen, 2002.

[10] Yngve Lamo and Michał Walicki. Modeling partiality by nondeterminism. In N. Callaos, J. M. Pineda, and M. Sanchez, editors, *Proceedings of SCI/ISAS 2001*, volume I. Orlando, FL, 2001.

[11] Aida Pliuškievičienė. Specialization of the use of axioms for deduction search in axiomatic theories with equality. *J. Soviet Math.*, 1, 1973.

[12] H. Rasiowa and R. Sikorski. *The Mathematics of Metamatemathics*. PWN [Polish Scientific Publishers], 1963.

[13] Michał Walicki and Sigurd Meldal. A complete calculus for the multialgebraic and functional semantics of nondeterminism. *ACM TOPLAS*, 17(2), March 1995.

[14] Michał Walicki and Sigurd Meldal. Multialgebras, power algebras and complete calculi of identities and inclusions. In *Recent Trends in Algebraic Specification Techniques*, volume 906 of *LNCS*. Springer, 1995.

[15] Michał Walicki and Sigurd Meldal. Algebraic approaches to nondeterminism-an overview. *ACM Computing Surveys*, 29, 1997.