Reasoning and Rewriting with Set-Relations I: Ground Completeness

Valentinas Kriaučiukas[1] and Michał Walicki[2] *

[1] Department of Mathematical Logic
Institute of Mathematics and Informatics, Vilnius, LITHUANIA
{valentinas.kriauciukas@mlats.mii.lt}
[2] Deptartment of Informatics, University of Bergen, NORWAY
{michal@ii.uib.no}

Abstract. The paper investigates reasoning with set-relations: intersection, inclusion and identity of 1-element sets. A language is introduced which, interpreted in a multialgebraic semantics, allows one to specify such relations. An inference system is given and shown sound and refutationally ground-complete for a particular proof strategy which selects only maximal literals from the premise clauses. Each of the introduced set-relations satisfies only two among the three properties of the equivalence relations - we study rewriting with such non-equivalence relations and point out differences from the equational case. As a corollary of the main ground-completeness theorem we obtain ground-completeness of the introduced rewriting technique.

1 Introduction

Reasoning with sets becomes an important issue in different areas of computer science. Its relevance can be noticed in constraint and logic programming e.g. [SD86, DO92, Jay92, Sto93], in algebraic approach to nondeterminism e.g. [Hus93, Hes88, WM95], in term rewriting e.g. [LA93, Kap88, Hus93].

Our interest in the set concepts originates from an earlier study of specifications of nondeterministic operations. Such operations are naturally modelled as set-valued functions. The semantic structures serving this purpose - *multialgebras* - generalize the traditional algebras allowing operations which, for a given argument, return not necessarily a single value but a set of values (namely, the set of all possible values returned by an arbitrary application of the operation). In [WM95, Wal93] we defined a specification language using set-relations and its multialgebraic semantics. The set-relations we considered were: inclusion, intersection and identity of 1-element sets. The first two are the usual set relations. Inclusion allows one to define set equality which, for that reason, is not included in the language. The third relation is particularly important: it provides the syntactic means of distinguishing between sets and their elements, and is indispensable for obtaining a complete reasoning system. Such a system is also given in the above works.

* Both authors gratefully acknowledge the financial support received from the Norwegian Research Council.

In the present paper we use the same set-relations but introduce a new reasoning system. It is less general than the earlier one - we are studying only the ground case - but it is much more prone to automation. Rewriting with non-congruence relations becomes also an issue of increasing importance. The set-relations we are considering are not even equivalences: equality is symmetric and transitive (but not reflexive), inclusion is reflexive and transitive (but not symmetric) and intersection is reflexive and symmetric (but not transitive). We study the rewriting proofs in the presence of these relations generalizing several classical notions (critical pair, confluence, rewriting proof) to the present context. Our results on rewriting extend bi-rewriting [LA93] in that we consider three different set-relations. We also take a step beyond the framework of [BG93] in that we study more general composition of relations than chaining of transitive relations.

Section 2 defines the syntax and the multialgebraic semantics of the language and lists some basic properties of the set-relations. Section 3 introduces the reasoning system $\mathcal{I}$, ordering of words and specifies the *maximal literal* proof strategy for using $\mathcal{I}$. Section 4 discusses term rewriting with the introduced set-relations. In section 5 we discuss the main theorem - refutational ground completeness of $\mathcal{I}$ with the maximal literal strategy and, as a simple corollary, ground completeness of rewriting.

The present paper is an improved and shortened version of the report [KW94]. Because of the space limitations, all the proofs had to be omitted in the present version of the paper.

2 Specifications of Set-Relations

Specifications are written using a finite set of function symbols $\mathcal{F}$ having arity $ar : \mathcal{F} \to \mathbb{N}$.[3] A symbol $f \in \mathcal{F}^0$ is called a *constant*. Only ground case is considered here, and we do not introduce any variables. We denote by $\mathcal{T}(\mathcal{F})$ the set of all (ground) terms. There are only three atomic forms using binary predicates: *equation* $s \approx t$, *inclusion* $s \prec t$ and *intersection* $s \frown t$. A *specification* is a set of *clauses* – finite sets of *literals*, where a *literal* is an atom or a negated atom written $\neg a$. (In [WM95, Wal93] a restricted language is used, allowing only negated intersections, positive inclusions and equations in clauses.) We will usually write negated atoms explicitly as $s \not\approx t$, $s \not\prec t$ and $s \not\frown t$, and assume $\neg(\neg a) = a$. By *words* we will mean the union of the sets of terms, literals and clauses. We will write $u[s]_p$ to denote that a term s is a subterm of a term u at a position p. Often the position will be omitted for the sake of simplicity.

Syntactic expressions of the language are interpretated in *multialgebras* [Kap88, Hus92, Wal93].

Definition 2.1 An $\mathcal{F}$-*multialgebra* A is a tuple $\langle S^A, \mathcal{F}^A \rangle$ where S^A is a non empty *carrier* set, and $\mathcal{F}^A$ is a set of set-valued functions $f^A : (S^A)^{ar(f)} \to$

[3] We are treating only the unsorted case – extension to many sorts is straightforward.

$\mathcal{P}^+(S^A)$, where $f \in \mathcal{F}$, and $\mathcal{P}^+(S^A)$ is the power-set of S^A with the empty set excluded.

Defining the meaning of the words we follow [WM95, Wal93].

Definition 2.2 An interpretation $[\![d]\!]^A$ of any expression d of the language is defined as follows:

- $[\![c]\!]^A \stackrel{\text{def}}{=} c^A$, if c is a constant;
- $[\![f(t_1, \ldots, t_n)]\!]^A \stackrel{\text{def}}{=} \bigcup\{f^A(\alpha_1, \ldots, \alpha_n) : \alpha_i \in [\![t_i]\!]^A\}$ for any $f \in \mathcal{F}^n$ and $\{t_1, \ldots, t_n\} \subset \mathcal{T}(\mathcal{F})$;
- $[\![s \approx t]\!]^A$ is true if $[\![s]\!]^A = [\![t]\!]^A = \{\alpha\}$ for some $\alpha \in S^A$, and false otherwise;
- $[\![s \prec t]\!]^A$ is true if $[\![s]\!]^A \subseteq [\![t]\!]^A$, and false otherwise;
- $[\![s \frown t]\!]^A$ is true if $[\![s]\!]^A \cap [\![t]\!]^A \neq \emptyset$, and false otherwise;
- for an atom a, $[\![\neg a]\!]^A$ is true if $[\![a]\!]^A$ is false, and is false otherwise;
- $[\![a_1, \ldots, a_n]\!]^A$ is true if some $[\![a_i]\!]^A$ is true, and false otherwise.

Definition 2.2 implies that for each $f \in \mathcal{F}$, f^A is $\subseteq$-monotone (because it is defined by pointwise extension). Interpretation of a constant c is, according to the definition of multialgebra, a non-empty set. Observe also that equality is not reflexive — $t \approx t$ is not true in general. A term t for which this equality is true is called *deterministic* because then it has only one possible value. The equality is merely a symmetric and transitive relation. An inclusion $s \prec t$ means that the term s has the value set which is included in the value set of t. This relation is a partial preorder — it is transitive and reflexive, but not symmetric. The intersection is reflexive (because of nonemptiness of term values) and symmetric, but lacks the transitivity property. Thus each of these relations satisfies two of the three properties of equivalence relations. Now we present some other properties of these relations.

2.1 Basic Properties of Literals

The following relation expresses equality of term value sets, and is the usual interpretation of equality in the set-valued approach to nondeterminism [Hes88, Kap88].

$$s \asymp t \stackrel{\text{def}}{=} s \prec t \wedge s \succ t. \tag{1}$$

As can be expected, it does not increase expressibility and therefore is not used in the language. For a discussion of the intended meaning and difference between '$\approx$' and '$\asymp$' in the context of nondeterminism see [WM95, Wal93].

The positive, resp. negative, relations are totally ordered by strength:

$$u \approx v \Rightarrow u \asymp v \Rightarrow u \prec v \Rightarrow u \frown v \qquad \textit{and} \qquad u \not\approx v \Leftarrow u \not\asymp v \Leftarrow u \not\prec v \Leftarrow u \not\frown v \tag{2}$$

The following two lemmas present the subterm replacement and composition (chaining) properties. Replacement of "equals by equals" occurs only in the case of two of the four relations. Nevertheless these properties will allow us later to develop techniques of term-rewriting.

Lemma 2.3 *The following properties hold for the introduced predicates:*
$s{\approx}t \Rightarrow u[s]_p{\asymp}u[t]_p$, $s{\asymp}t \Rightarrow u[s]_p{\asymp}u[t]_p$, $s{\prec}t \Rightarrow u[s]_p{\prec}u[t]_p$, $s{\frown}t \Rightarrow u[s]_p{\frown}u[t]_p$.

Lemma 2.4 *The predicates satisfy the composition properties, given in Table 1.*

	$s \approx u$	$s \prec u$	$s \succ u$	$s \frown u$	$s \not\approx u$	$s \not\prec u$	$s \not\succ u$	$s \not\frown u$
$s \approx t$	$t \approx u$	$t \prec u$	$t \approx u$	$t \prec u$	$t \not\succ u$	$t \not\frown u$	$t \not\succ u$	$t \not\frown u$
$s \prec t$	$t \succ u$	$t \frown u$	$t \succ u$	$t \frown u$	$t \not\approx u$	$t \not\prec u$	$t \not\approx u$	$t \not\prec u$
$s \succ t$	$t \approx u$	$t \prec u$	–	–	–	–	$t \not\succ u$	$t \not\frown u$
$s \frown t$	$t \succ u$	$t \frown u$	–	–	–	–	$t \not\approx u$	$t \not\prec u$

Table 1. Rules for atom composition

For convenience we will write the partial function coded in this table as $\oplus \diamond \otimes = \odot$, meaning that $\odot$ is the strongest relation obtained by composing $\oplus$ and $\otimes$ for any terms, *i.e.*: $\oplus \diamond \otimes = \odot \iff (s \oplus t \wedge t \otimes u \Rightarrow s \odot u)$ for any terms s, t, u. Note that the table defines only the strongest composite of the arguments. Because of the ordering (2) the fact that, for instance, $\approx \diamond \not\frown = \not\frown$ will imply that also $\not\prec$ can be obtained from composing $\approx$ and $\not\frown$.

Composition of negative and positive atoms is symmetric to the composition of the positive and the negative ones given in the table. Composition of two negative atoms does not allow one to draw any conclusion and therefore is not mentioned at all.

Lemma 2.5 *The composition function* $\_ \diamond \_$ *is transitive.*

The next lemma is an easy corollary of the two previous lemmas, but it is important because it describes the situation known from term rewriting as a *critical peak* [DJ90] and is related with generation of *critical pairs*.

Lemma 2.6 *The atoms satisfy the replacement rules from Table 2.*

	$u[s] \approx v$	$u[s] \prec v$	$u[s] \succ v$	$u[s] \frown v$	$u[s] \not\approx v$	$u[s] \not\prec v$	$u[s] \not\succ v$	$u[s] \not\frown v$
$s \approx t$	$u[t] \approx v$	$u[t] \prec v$	$u[t] \succ v$	$u[t] \frown v$	$u[t] \not\approx v$	$u[t] \not\prec v$	$u[t] \not\succ v$	$u[t] \not\frown v$
$s \prec t$	$u[t] \succ v$	$u[t] \frown v$	$u[t] \succ v$	$u[t] \frown v$	$u[t] \not\approx v$	$u[t] \not\prec v$	$u[t] \not\approx v$	$u[t] \not\prec v$
$s \succ t$	$u[t] \approx v$	$u[t] \prec v$	–	–	–	–	$u[t] \not\succ v$	$u[t] \not\frown v$
$s \frown t$	$u[t] \succ v$	$u[t] \frown v$	–	–	–	–	$u[t] \not\approx v$	$u[t] \not\prec v$

Table 2. Rules for term replacement in atoms

The content of this table is also encoded as a partial function: $\mathsf{Repl}(\oplus, \otimes) = \odot \iff (s \oplus t \wedge u[s]_p \otimes v \Rightarrow u[t]_p \odot v)$ for any terms s, t, u, v and position p at u.

The two tables differ in predicate signs at four places — 1:3 - 1:6 (row 1, columns 3 through 6), where the relation resulting from Table 1 is stronger than the one from Table 2. These cases must be distinguished when the superposition rule (see below) is applied. Therefore we introduce the function $\mathsf{Sup}(s, t, \oplus, \otimes)$, which will select the appropriate table. Its value is $\oplus \diamond \otimes$ in the case $s = t$, and $\mathsf{Repl}(\oplus, \otimes)$, otherwise.

3 The Inference System

The following set of rules was constructed in analogy to the inference systems for first-order predicate calculus with equality [BG91, S-A92]. However, there are some additional restrictions due to the composition laws as compared with the equational case. Very similar rules are presented in [BG93] for transitive relations.

Reflexivity resolution $\dfrac{C, s \oplus s}{C}$, where $\oplus \in \{\not\prec, \not\succ, \not\frown\}$.

Superposition $\dfrac{C, s \oplus t \qquad D, u[s]_p \otimes v}{C, D, u[t]_p \odot v}$, where $\odot = \mathsf{Sup}(s, u[s]_p, \oplus, \otimes)$.

Compositionality resolution $\dfrac{C, s \oplus t \qquad D, s \odot u}{C, t \otimes u, s \odot u}$, where $\odot = \oplus^{-1} \diamond \neg\otimes$.

The analogous rule for equality called *equality factoring* [BG91, S-A92] is a special case of our rule when both premise clauses coincide. In [BG93] analogous rule is called *transitivity resolution*.

Let $\mathcal{I}$ denote the inference system consisting of the above rules.

Theorem 3.1 *The inference system $\mathcal{I}$ is sound.*

Proof. Soundness of reflexivity resolution follows from reflexivity of '$\prec$' and '$\frown$'. Soundness of superposition is a direct consequence of the replacement and composition laws (Lemmas 2.6, 2.4). Soundness of the compositionality rule is based on the following short deduction. Suppose that the first premise clause and the implication $s \oplus t \wedge \neg(t \otimes u) \Rightarrow s \odot u$ both are true. The later is equivalent to $s \oplus t \Rightarrow t \otimes u \vee s \odot u$. A single application of the (usual) resolution rule gives the conclusion of the rule. The second premise clause is not used in this step — it only shows the goal atom.

3.1 Ordering of Words

Various orderings of terms and atoms are used extensively in the study of automated deduction. We will apply such an ordering to define a more specific proof strategy for the system $\mathcal{I}$, to study the possibility of rewriting wrt. the introduced predicates and, finally, to define the model in the completeness proof. We assume the existence of a *simplification ordering* '$>$' [DJ90] on ground terms which is *total* ($\forall s \neq t \in \mathcal{T}(\mathcal{F}) : s > t \vee t > s$), *well-founded* ($\forall t \in \mathcal{T}(\mathcal{F}) : \{s : s < t\}$

is finite), *monotone* ($\forall u, s, t \in \mathcal{T}(\mathcal{F}) : s > t \Rightarrow u[s] > u[t]$) and *increasing* ($\forall s \in \mathcal{T}(\mathcal{F}) : u[s] \neq s \Rightarrow u[s] > s$). Ordering of other words is defined by the *multiset extension* [DM79] of this ordering. Let $\mathcal{M}(T)$ denote the set of all finite multisets of elements from T. Each element of $\mathcal{M}(T)$ can be represented by a function $\beta : T \to \mathbb{N}$ such that $\beta \equiv 0$ except for some finite number of elements of T. $\beta(d)$ is a number of copies of d in the multiset β.

Definition 3.2 For an ordering '$\rightarrow$' on a given set T, an ordering '$\rightarrow^m$' on the set $\mathcal{M}(T)$ is a *multiset extension* of '$\rightarrow$', if

$$\beta \rightarrow^m \gamma \iff \forall d \in T\, \exists c \in T\, ((\beta(c) > \gamma(c) \wedge (\beta(d) \geq \gamma(d) \vee c \rightarrow d)).$$

In the particular case of total ordering of T, which is the only one considered here, $\alpha \rightarrow^m \beta$ means that there is some $c \in T$ such that: $\alpha(c) > \beta(c) \wedge \forall d \rightarrow c\, \alpha(d) = \beta(d)$. This is a lexicographic ordering comparing biggest components first. In the general case it is known [DM79] that '$\rightarrow^m$' is total if '$\rightarrow$' is total and '$\rightarrow^m$' is well-founded if '$\rightarrow$' is well-founded.

Writing a literal $s \oplus t$, we indicate that $s \geq t$. It explains why both signs '$\prec$' and '$\succ$' are used. This rule, of course, is not applied to the conclusions of the proof rules. We assume that any term is bigger than any predicate symbol. A stronger positive predicate is bigger than a weaker one, the order between negative predicates is reversed, and all negative predicates are bigger than the positive ones:

$$\not\approx \;>\; \not\prec \;>\; \not\succ \;>\; \not\frown \;>\; \approx \;>\; \prec \;>\; \succ \;>\; \frown. \tag{3}$$

By analogy with the commonly used approach in equational reasoning, we identify literals with multisets. A literal $s \oplus t$ is represented by the multiset $\{\{s, \oplus\}, \{t, \oplus^{-1}\}\}$. The ordering of the predicates will make the negated form of an atom bigger than the atom itself.

The ordering of literals is the twofold extension of '$<$' because each literal is a multiset of two multisets. The biggest literal in a clause C w.r.t. this ordering is denoted by $\max(C)$. Clauses are compared as multisets of literals, so their ordering is the multiset extension of the ordering of literals (threefold multiset extension of '$<$'). Although we have here three different orderings, we will use the same symbol '$<$' to denote any of them. This should not introduce any confusion as the sets of terms, literals and clauses are disjoint.

3.2 The MAXIMAL LITERAL Proof Strategy

The literals mentioned explicitly in the premises of the proof rules are called *active*. Various ways of selecting the active literals will lead to different proof strategies. The *maximal literal* strategy requires that the active literals in the premise clauses are the ones which are maximal wrt. the ordering defined above. Stated explicitly the strategy amounts to the following restrictions on the application of the rules:

Reflexivity resolution: the literal $s \oplus s$ is maximal in the premise clause.

Superposition: the atom $s \oplus t$ and the literal $u[s]_p \otimes v$ are maximal in their clauses.

Compositionality resolution: the atom $s \oplus t$ is maximal in its clause. The atom $s \odot u$ **is not** maximal in the second premise clause, but the term s **is** maximal in this clause, and the maximal literal in this clause is positive. Both $\oplus$ and $\odot$ are positive.

The restriction on the last rule is the only case where some active atom $(s \odot u)$ is not maximal in its clause. However, it is almost maximal because the maximal term s of the clause occurs in it. Another reason why this weakening of the strategy is not essential is that the second clause in the premise provides merely the context allowing application of the rule, and in fact the atom $s \odot u$ is not so "active". A particular consequence of this restriction is that the rule can be applied only when its second premise is a non-Horn clause.

4 Rewriting Proofs

In the next section we show that if the empty clause can not be deduced using the maximal literal strategy, then a model exists satisfying the initial set of clauses. The model is constructed from an appropriate set of ground atoms which force all the initial clauses to be true. The notion of forcing requires construction of a deductive closure of a given set of literals. This section investigates the rewriting proofs in which ground literals are rewritten to ground literals. The obtained results will serve as a basis for the construction of forcing set in the completeness proof.

Although, eventually, only atoms will be used in the model construction we give a more general account – our definitions and lemmas apply to rewriting with both negative and positive literals.

Rewriting of literals with the set-relations is based on the fact that the relations satisfy replacement properties from Lemma 2.6. For example, the implication: $s \frown t \Rightarrow u[s]_p \frown u[t]_p$ means that the atom $u[s]_p \frown u[t]_p$ can be derived applying the rule $s \stackrel{\frown}{\longrightarrow} t$ to the term $u[s]_p$. The following definition states what kind of literals can be derived directly applying the replacement property to some set $\mathcal{A}$ of ground literals (also called *axioms*).

Definition 4.1 A literal r is a *rewriting step* in $\mathcal{A}$ if either 1) $r \in \mathcal{A}$, or 2) r is an atom $u[s]_p \oplus u[t]_p$, and $s \otimes t \in \mathcal{A}$, where $\oplus = \otimes$, if $\otimes \neq \approx$, or $\oplus \in \{\prec, \succ\}$, otherwise.

The rule based directly on this kind of term-rewriting is superposition. The forcing set in the completeness proof will consist of ground positive literals, which can be derived using only this one rule. The superposition rule takes two rewriting steps and composes them into one. Consequtive applications of the superposition correspond to composition of the finite sequence of the corresponding rewriting steps $\langle s \oplus_1 t_1, t_1 \oplus_2 t_2, t_2 \oplus_3 t_3, \ldots, t_n \oplus_n t \rangle$. Such a sequence is called a *rewriting sequence*, and the predicate sign $\oplus$ of the resulting literal is computed using the function $\_\diamond\_$: $\oplus = ((\oplus_1^{-1} \diamond \oplus_2)^{-1} \diamond \cdots)^{-1} \diamond \oplus_n$. The next definition puts all such literals into *rewriting closure* of $\mathcal{A}$. This closure also contains atoms that are trivially true.

Definition 4.2 For a set $\mathcal{A}$ of ground literals, the *rewriting closure* of $\mathcal{A}$ is the set of ground literals, $\mathcal{A}^*$, defined as follows:

- all atoms of the form $s \prec s$ or $s \frown s$ belong to $\mathcal{A}^*$;
- if an atom $s \oplus t \in \mathcal{A}$ and a literal $u[s] \otimes v \in \mathcal{A}^*$, then the literal $u[t] \odot v \in \mathcal{A}^*$, if $\odot = \mathsf{Sup}(s, u[s], \oplus, \otimes)$.

Primarily, we are interested in *reducing* rewriting sequences, *i.e.*, such that rewriting is used to produce terms of lower complexity in some well-founded ordering. The term ordering is used to orient literals but it does not allow us to orient the *reflexive* literals of the form $s \oplus s$. However, the orientation problem of these particular literals turns out to be inessential for the following arguments (except the next definition), and so we allow them to have orientation that is appropriate for the context in which it is used. A literal $s \oplus t$ can be written in the form $s \xrightarrow{\oplus} t$ to emphasize that $s \geq t$, then it is also called a *rule*. The fact that this literal is derived by a rewriting sequence in which terms do not increase in any step is written as $s \stackrel{\oplus}{\Longrightarrow} t$ or (the same) as $t \stackrel{\oplus^{-1}}{\Longleftarrow} s$.

Definition 4.3 A rewriting sequence is *reducing* (w.r.t. to an ordering of terms $<$) if it does not contain a *peak*, *i.e.*, a pair of consecutive rewriting steps $s \oplus t, t \otimes u$ such that $s \leq t \geq u$. A *rewriting proof* is a reducing rewriting sequence.

The non-strong inequalities in the last condition capture the cases of reflexive steps. A rewriting step $s \oplus s$ does not form a peak in a rewriting sequence only at a locally minimal point, *i.e.*, in a rewriting proof where s is the smallest term. Definition 4.3 means that any reducing proof consists of two decreasing branches like $s \Longrightarrow u \Longleftarrow t$, or has only one $s \Longrightarrow t$ or $s \Longleftarrow t$. The table from Lemma 2.4 can be written as a summary of all the possible combinations of the resulting predicate signs appearing in two-branches rewriting proofs:

$$\begin{array}{rcl}
(s \stackrel{\approx}{\Longrightarrow} u \stackrel{\approx}{\Longleftarrow} t \vee s \stackrel{\approx}{\Longrightarrow} u \stackrel{\succ}{\Longleftarrow} t \vee s \stackrel{\prec}{\Longrightarrow} u \stackrel{\approx}{\Longleftarrow} t) & \Rightarrow & s \approx t, \\
(s \stackrel{\approx}{\Longrightarrow} u \stackrel{\prec}{\Longleftarrow} t \vee s \stackrel{\approx}{\Longrightarrow} u \stackrel{\frown}{\Longleftarrow} t \vee s \stackrel{\prec}{\Longrightarrow} u \stackrel{\prec}{\Longleftarrow} t) & \Rightarrow & s \prec t, \\
(s \stackrel{\succ}{\Longrightarrow} u \stackrel{\succ}{\Longleftarrow} t \vee s \stackrel{\succ}{\Longrightarrow} u \stackrel{\approx}{\Longleftarrow} t \vee s \stackrel{\frown}{\Longrightarrow} u \stackrel{\approx}{\Longleftarrow} t) & \Rightarrow & s \succ t, \\
(s \stackrel{\succ}{\Longrightarrow} u \stackrel{\frown}{\Longleftarrow} t \vee s \stackrel{\succ}{\Longrightarrow} u \stackrel{\prec}{\Longleftarrow} t \vee s \stackrel{\frown}{\Longrightarrow} u \stackrel{\prec}{\Longleftarrow} t) & \Rightarrow & s \frown t.
\end{array}$$

Lemma 2.6 describes how the peaks can be eliminated from rewriting sequences. Let us take, for example, one implication from this lemma: $s \frown t \wedge u[s] \approx v \Rightarrow u[t] \succ v$. The premise can be interpreted as a possibility to have a peak $u[t] \xleftarrow{\frown} u[s] \xrightarrow{\approx} v$ in proofs, if both atoms $s \frown t$ and $u[s] \approx t$ are axioms. This peak can be "cut down" changing it by the consequence $u[t] \succ v$, if it is also among the axioms. The following notions are commonly used in similar situations.

Definition 4.4 A rule $r_1 = s \xrightarrow{\oplus} t$ *overlaps* a rule $r_2 = u[s]_p \xrightarrow{\otimes} v$. In this case the literal $l = u[t]_p \odot v$, where $\odot = \mathsf{Sup}(s, u[s]_p, \oplus, \otimes)$, is called a *critical literal* formed by the rules r_1, r_2, if l is different from r_1 and r_2.

Critical literals correspond to *critical pairs* from equational reasoning [DJ90]. In our case the definition is more complex because the predicate sign is important and replacement is not merely of "equals by equals". Also, when the rule r_1 is reflexive (which is not necessarily a tautology in our case) then the critical literal l may be the same as r_2. It is better to exclude such cases because they would complicate our model construction.

Definition 4.5 A set $\mathcal{R}$ of ground rewriting rules is *confluent* if $\mathcal{R}^*$ contains all critical literals formed by overlapping rules from $\mathcal{R}$.

In term-rewriting theory [DJ90] such systems are called *locally-confluent*. Confluent systems have slightly different definition, but both these notions are proved to be equivalent. In [LA93] a similar definition introduces bi-confluent systems.

In completeness proofs like ours, *fully-reduced* [PP91] or *left-reduced* [S-A92, BG91] rewriting systems are used. We are not able to define the analogous notion, since deduction and reduction are not the same in our language, and will apply Definition 4.5 instead. Its direct consequence is

Lemma 4.6 *Any literal derivable by a rewriting sequence in a confluent system $\mathcal{R}$ has a rewriting proof in $\mathcal{R}$.*

Since we have allowed both negative and positive literals to occur in one set of axioms $\mathcal{A}$, the unpleasant situation, when both an atom a and its negation $\neg a$ belong to $\mathcal{A}^*$, is possible. The set $\mathcal{A}$ is *consistent* if no such atom exists. A set containing only atoms is obviously consistent. Although such a set will be of main importance in the following section, we again formulate stronger results, taking into account the general situation of possibly inconsistent sets of literals. Next lemma, to be used in the completeness proof to construct confluent systems incrementally, characterizes rules that can be added to a confluent and consistent system preserving both these properties. Here we have again the situation different from the usual equational reasoning, because the rule $s \approx t$ overlaps itself and produces the critical atom $t \approx t$ which need not be always true. The following lemma characterizes the rules which can be added to a system preserving its consistency and confluence. It serves as a basis for the construction of the forcing set in the next section.

Lemma 4.7 *For a confluent and consistent system $\mathcal{R}$ and a rule $r \notin \mathcal{R}^*$ the system $\mathcal{R} \cup \{r\}$ is confluent and consistent iff*

(i) *r does not have the form $s \xrightarrow{\oplus} s$, where $\oplus \in \{\not<, \not>, \not\wedge\}$,*

(ii) *for any critical literal l formed by any $r' \in \mathcal{R} \cup \{r\}$ overlapping (or overlapped by) r, $l \in \mathcal{R}^*$.*

5 The Completeness Theorem

The proof system $\mathcal{I}$ is used to derive a clause from a given specification $\mathcal{S}$ "by contradiction": to prove that a clause $C = \{a_1, \ldots, a_n\}$ follows from $\mathcal{S}$, one takes the negation of C, namely the set of unary clauses $neg(C) \stackrel{\text{def}}{=} \{\neg a_1; \ldots; \neg a_n\}$, adds it to $\mathcal{S}$, and tries to derive the empty clause from the resulting set of clauses. Proving *refutational completeness*, we have to show that if some set of ground clauses $\mathcal{S}$ has no model, then the empty clause is derivable using rules from $\mathcal{I}$. The usual way to prove this is to show that there exists a model satisfying all the clauses from $\mathcal{S}$ if the empty clause is not derivable from $\mathcal{S}$. In our proof we follow the ideas from [S-A92] and [BG93] which, in turn, develop the ideas of [Bez90]. Similar proof using forcing is given in [PP91]. All these works are concerned with first-order predicate calculus with equality. In [BG93], a similar proof method is used with respect to transitive relations.

Our construction proceeds in two main steps. Given a consistent set $\mathcal{S}$ of clauses, we select a set of atoms $\mathcal{R}$ (section 5.1) and show that $\mathcal{R}$ is a *forcing set* for the clauses from $\mathcal{S}$. Then (section 5.2) we show that $\mathcal{R}$ can be used to construct a multimodel which satisfies $\mathcal{S}$.

We call a set of clauses $\mathcal{S}$ *consistent* if it does not contain the empty clause. The *redundancy* of clauses in $\mathcal{S}$ will be defined during the model construction. Redundancy notion was developed by Bachmair and Ganzinger [BG91] to cover simplification techniques commonly used in theorem provers. Referring to this notion we fix a set $\mathcal{S}$ and assume it is consistent and *relatively closed*, meaning that any application of a rule from $\mathcal{I}$ with premises from $\mathcal{S}$ produces a clause that is in $\mathcal{S}$ or is redundant in $\mathcal{S}$. The main result is

Theorem 5.1 (Ground-completeness) *If a set of ground clauses $\mathcal{S}$ is consistent and relatively closed then it has a model.*

In the following sections we merely indicate the main steps and results needed in the proof of this main theorem.

5.1 Forcing Set, Redundancy and Productive Clauses

We borrow the notion of forcing from [PP91] where it is also used in a completeness proof:

Definition 5.2 A set of ground atoms $\mathcal{A}$ *forces*

- a ground atom a if $a \in \mathcal{A}^*$, and the literal $\neg a$ if $a \notin \mathcal{A}^*$;
- a clause C if it forces some literal $l \in C$;
- a set of clauses $\mathcal{S}$ if it forces all clauses from $\mathcal{S}$.

In the last case we say that $\mathcal{A}$ is a *forcing set* for $\mathcal{S}$. We write $\mathcal{A} \Vdash w$ if $\mathcal{A}$ forces w. For a consistent set $\mathcal{S}$ of ground clauses we will construct a set $\mathcal{R}$ of ground atoms forcing $\mathcal{S}$. All such atoms can be oriented into rules because of our assumption about an ordering of terms, therefore we can treat $\mathcal{R}$ as a system of rules. Since the constructed $\mathcal{R}$ will be confluent, it suffices to consider only rewriting proofs.

The starting point of the model construction is the set of maximal literals

$$\mathcal{A}_0 \overset{\text{def}}{=} \{\max(C) : C \in \mathcal{S}\}. \tag{4}$$

In rewriting proofs all terms are not bigger than the maximal term of the literal being proved. This admits an incremental construction of the model, starting with $\mathcal{A}_0$ and removing redundant literals.

Redundancy of clauses is defined relatively to two sets: one set of clauses $\mathcal{S}$ and one of ground atoms $\mathcal{A}$. This is an intermediate notion, the final one refers only to $\mathcal{S}$. We have already fixed the set of clauses $\mathcal{S}$ to shorten our formulations. For a given literal l and a set $\mathcal{L}$ of literals the set $\mathcal{L}_l \overset{\text{def}}{=} \{a \in \mathcal{L} : a < l\}$ contains all the literals from $\mathcal{L}$ that are smaller than l.

Definition 5.3 A clause $C \in \mathcal{S}$ with $\max(C) = l$ is *redundant* in a set of ground atoms $\mathcal{A}$ if either

- $\mathcal{A}_l \Vdash C$ or
- $\mathcal{S}$ contains another clause $C' < C$ with $\max(C') = l$, such that $\mathcal{A}_l \nVdash C'$.

The nature of the second condition of the definition may not be very clear, but thanks to this condition, the whole definition is a negated assertion about some minimality of a clause. Statements of this kind are very appropriate in inductive proofs, like our proof of completeness. The redundancy of literals is based on redundancy of clauses and Lemma 4.7.

Definition 5.4 A ground literal l is *redundant* in a set of ground atoms $\mathcal{A}$ if either

- $l = s \oplus s$, where $\oplus \in \{\not\prec, \not\succ, \not\frown\}$, or
- $\mathcal{A} \cup \{l\}$ contains a rule r overlapping l and forming the critical literal a, such that $\mathcal{A} \nVdash a$, or
- every clause $C \in \mathcal{S}$ with $\max(C) = l$ is redundant in $\mathcal{A}$.

We write $\mathsf{red}(\mathcal{A}, w)$ to indicate that w is redundant in $\mathcal{A}$. Observe that Definitions 5.4 and 5.2 of redundancy and forcing are so related, that all negative literals that are not forced are redundant. Since any forced literal makes all clauses containing it redundant, any negative literal appears redundant.

After all the preliminary definitions the definition of the forcing set is quite short. The set is defined as a limit of a decreasing sequence of sets, which begins with $\mathcal{A}_0$ defined in (4). Succeeding sets are obtained removing minimal redundant literals. Suppose $\mathcal{A}_i$ is already known, and let l_i be the minimal redundant literal in $\mathcal{A}_i$:

$$\mathcal{A}_{i+1} \overset{\text{def}}{=} \mathcal{A}_i \setminus \{l_i\}, \qquad\qquad \mathcal{R} \overset{\text{def}}{=} \bigcap_{i \in \mathbf{N}} \mathcal{A}_i. \tag{5}$$

The next lemma shows that redundancy is preserved when taking the limit in the definition of $\mathcal{R}$, and that redundancy of a word in some $\mathcal{A}_i$ is equivalent to its redundancy in $\mathcal{R}$.

Lemma 5.5 *For a literal* $l \in \mathcal{A}_0$ *(a clause* C *with* $\max(C) = l$*) :*

$$\exists i, l_i > l : \mathsf{red}(\mathcal{A}_i, l) \iff (\forall j > i : \mathsf{red}(\mathcal{A}_j, l) \iff \mathsf{red}(\mathcal{R}, l)).$$

As an immediate consequence of this lemma and Lemma 4.7 we obtain

Corollary 5.6 $\mathcal{R}$ *is confluent.*

With every atom a from $\mathcal{R}$ there is associated a clause from $\mathcal{S}$ which causes a to be included in $\mathcal{R}$. In [S-A92] such clauses are called *regular*, in [BG91] *productive* because they produce atoms being included in the forcing set. In [PP91], where the forcing method is presented, no special notion for clauses of this kind is used.

Definition 5.7 A clause C, l with $\max(C, l) = l$ is *productive* for l in a set $\mathcal{A}$ if $\mathcal{A} \not\Vdash C$.

The main properties of the set $\mathcal{R}$ are expressed in the following theorem:

Theorem 5.8 *Let* $\mathcal{S}$ *be consistent and relatively closed set of ground clauses,* $\mathcal{A}_0$ *and* $\mathcal{R}$ *be as defined by (4) and (5). Any literal* $l \in \mathcal{A}_0$ *satisfies the following conditions:*

I1. *if* $\neg\mathsf{red}(\mathcal{R}, l)$*, then for any* $a \in \mathcal{R}_l$ *there exists a clause* $C \in \mathcal{S}$ *productive for* a *in* $\mathcal{R}_l$*,*

I2. *if* $\mathsf{red}(\mathcal{R}, l)$*, then for any clause* $C \in \mathcal{S}$ *with* $\max(C) = l : \mathcal{R}_l \Vdash C$*.*

The theorem is formulated in the form of an induction statement. After taking the limit $\mathcal{R} = \bigcup_{l \in \mathcal{A}_0} R_l$, the theorem means that any atom in $\mathcal{R}$ has productive clause in $\mathcal{S}$ (this is an auxiliary assertion) and that $\mathcal{R} \Vdash \mathcal{S}$. This is the main technical result of this paper.

5.2 From the Forcing Set to a Multialgebra

Thus, we have shown that for a consistent and relatively closed set $\mathcal{S}$ of ground clauses, the set of ground atoms $\mathcal{R}$ is a model of $\mathcal{S}$ in the sense that it forces all the clauses from $\mathcal{S}$. To complete the proof of Theorem 5.1 we need to show that the existence of such an $\mathcal{R}$ implies the existence of a multialgebra A which satisfies all the atoms from $\mathcal{R}^*$ and only these ones. Then, from the definition of forcing it follows that A also satisfies all the clauses $\mathcal{S}$.

The rewriting closure $\mathcal{R}^*$ defines a reflexive transitive relation '$\prec$' on the set of terms $\mathcal{T}(\mathcal{F})$. In multialgebras this partial (pre)order '$\prec$' on terms is interpreted as set inclusion: an atom $s \prec t$ means that $[\![s]\!]^A \subseteq [\![t]\!]^A$. Two other predicates of our language also have natural interpretation in partial order terms:

- $s \approx t$ means that both sets $[\![s]\!]^A$ and $[\![t]\!]^A$ are equal *minimal* elements in the partial order of nonempty sets, *i.e.*, they denote the same set *with one element*;
- $s \frown t$ means that there exists some minimal element α such that $\alpha \in [\![s]\!]^A$ and $\alpha \in [\![t]\!]^A$.

The relation '$\prec$' on the set $\mathcal{T}(\mathcal{F})$ is partial preorder, because different terms may have the same value set. To turn it into a partial order we have to take the quotient of $\mathcal{R}^*$ modulo '$\asymp$' that was defined in (1). Since $\asymp$ denotes set equality, it is obviously a congruence: reflexivity and transitivity follow from the analogous properties of inclusion '$\prec$', symmetricity from (1), and the replacement property is given in (2.3).

First, given a set $\mathcal{R}$ of ground atoms, we construct a partially ordered set $PO(\mathcal{R}) = \langle \mathcal{C}, \sqsubseteq \rangle$. Then, we extend signature with new constants and $\mathcal{R}$ with new atoms, so that $PO(\mathcal{R})$ defines a multialgebra satisfying all the atoms in $\mathcal{R}$.

Let $\mathcal{C} \stackrel{\text{def}}{=} \mathcal{R}^*/_{\asymp}$ be the quotient of $\mathcal{R}^*$ modulo '$\asymp$'. $[t] \stackrel{\text{def}}{=} \{s \in \mathcal{T}(\mathcal{F}) : s \prec t \in \mathcal{R}^* \wedge t \prec s \in \mathcal{R}^*\}$ denotes the equivalence class in $\mathcal{C}$ of a term t, and $\sqsubseteq \stackrel{\text{def}}{=} \{\langle [s], [t] \rangle : s \prec t \in \mathcal{R}^*\}$ is a partial order on $\mathcal{C}$ ('$\sqsubset$' is the irreflexive part of this relation). (That $\langle \mathcal{C}, \sqsubseteq \rangle$ is well defined, i.e, that all atoms in one equivalence class stand in the same relations, follows from Table 1: for any atom $s \oplus t$, any $t \prec u$ or $t \succ u$ is enough to derive also $s \oplus u$.) Let $\mathcal{D} \stackrel{\text{def}}{=} \{[s] : s \in \mathcal{T}(\mathcal{F}),\ s \approx s \in \mathcal{R}^*\}$ be the set of *deterministic* elements in $\mathcal{C}$, $\mathcal{D}(T) \stackrel{\text{def}}{=} \{S \in \mathcal{D} : S \sqsubseteq T\}$ be the set of deterministic elements which are smaller than or equal to $T \in \mathcal{C}$, and $\mathcal{M} \stackrel{\text{def}}{=} \{S \in \mathcal{C} : \forall T \in \mathcal{C}\ T \not\sqsubset S\}$ be the set of *minimal* elements in $\mathcal{C}$. $\mathcal{M}(S)$ denotes the set of elements from $\mathcal{M}$ that are smaller than or equal to S. We will write $\mathcal{M}[t]$ (or $\mathcal{D}[t]$) instead $\mathcal{M}([t])$ (respectively, $\mathcal{D}([t])$).

The set $\mathcal{D}$ is a subset of $\mathcal{M}$, because from $s \approx s$ and $s \succ u$ it follows that $s \approx u$, meaning that no elements lie below the class $[s]$ if $[s] \in \mathcal{D}$. The set $\mathcal{D}$ is a "candidate" to be a carrier of a multialgebra, and $\mathcal{D}(T)$ should be an interpretation mapping defining the multialgebra. Definition 2.2 tells us what properties $\mathcal{D}[\_]$ should have in order to be an interpretation mapping:

PO1. $\mathcal{D}[f(t_1, \ldots, t_n)] = \bigcup \{\mathcal{D}[f(\alpha_1, \ldots, \alpha_n)] : \alpha_i \in \mathcal{D}[t_i]\}$
for any $f \in \mathcal{F}^n$, $\{t_1, \ldots, t_n\} \subset \mathcal{T}(\mathcal{F})$;

PO2. $\mathcal{D}[s] = \{[s]\} \iff s \approx s \in \mathcal{R}^*$;

PO3. $\mathcal{D}[s] \subseteq \mathcal{D}[t] \iff s \prec t \in \mathcal{R}^*$;

PO4. $\mathcal{D}[s] \cap \mathcal{D}[t] \neq \emptyset \iff s \frown t \in \mathcal{R}^*$.

If $\mathcal{D}[\_]$ satisfies PO1–PO4, then a multialgebra A satisfying $\mathcal{R}$ can be defined:

MA1. the carrier $S^A \stackrel{\text{def}}{=} \mathcal{D}$;

MA2. for any constant $c \in \mathcal{F}$: $c^A \stackrel{\text{def}}{=} \mathcal{D}[c]$;

MA3. for any $f \in \mathcal{F}^n$ and $\{[d_1], \ldots, [d_n]\} \subseteq \mathcal{D}$:
$f^A([d_1], \ldots, [d_n]) \stackrel{\text{def}}{=} \mathcal{D}[f(d_1, \ldots, d_n)]$.

In this case $[\![\_]\!]^A = \mathcal{D}[\_]$, what follows from PO1, and we say that $\mathcal{R}$ *defines* the multialgebra A. The next result is important, because the multialgebra is constructed from positive atoms only, while clauses from $\mathcal{S}$ may also contain negative atoms. We must be sure that the multialgebra makes true only atoms derivable from the forcing set $\mathcal{R}$.

Lemma 5.9 *If $\mathcal{R}$ defines a multialgebra A then, for any atom $a : a \in \mathcal{R}^* \iff [\![a]\!]^A$ is true.*

However, the mapping $\mathcal{D}[\_]$ defined from the forcing set $\mathcal{R}$ may violate any of the Requirements PO1–PO4. We might therefore consider the mapping $\mathcal{M}[\_]$ instead, but also this one may violate these requirements. To meet these problems we have to extend $\mathcal{C}$ (and mapping $\mathcal{D}[\_]$) with new minimal elements. For instance, there may be no (deterministic) element validating the atom $s \frown t \in \mathcal{R}^*$, or even no such term included in a given term t (which therefore would denote empty set). We do not give here the, rather elaborate, details of this extension. Its result is that the signature is extended with new constants, and new atoms are added imposing the required properties on these new elements. In particular, all new elements are deterministic, which makes the mappings $\mathcal{D}[\_]$ and $\mathcal{M}[\_]$ coincide. The crucial property of this extension is that it is conservative.

Definition 5.10 Let $\mathcal{F} \subset \mathcal{F}_1$ be two signatures and $\mathcal{R}, \mathcal{R}_1$ two sets of atoms over signature $\mathcal{F}$, resp. $\mathcal{F}_1$. $\mathcal{R}_1$ is a *conservative extension* of $\mathcal{R}$ if for any $\mathcal{F}$-atom a, $a \in \mathcal{R}_1^* \iff a \in \mathcal{R}^*$.

Thus, we can complete the set $\mathcal{R}$ with elements and atoms needed to construct a multialgebra satisfying exactly the same atoms which are members of $\mathcal{R}$. Since $\mathcal{R} \Vdash \mathcal{S}$, this means that we obtain a multialgebraic model of $\mathcal{S}$. This last step of the construction is expressed in:

Theorem 5.11 *For any atom set $\mathcal{R}$ there exists an atom set $\mathcal{R}_1$ that is a conservative extension of $\mathcal{R}$ and defines a multialgebra.*

This ends the proof of the completeness theorem which also yields:

Corollary 5.12 *Any ground atom valid in all multimodels of a given set of ground atoms $\mathcal{R}$ has a rewriting proof in $\mathcal{R}$.*

6 Conclusion

Motivated by the study of nondeterministic specifications, we have introduced a system for reasoning with the set-relations: inclusion, intersection and identity of 1-element sets. The system gives rise to a rewriting technique where atoms involving these relations are rewritten to other atoms, and chaining is based on the explicitly specified composition laws for the introduced relations. We have shown that the reasoning system and, consequently, the associated rewriting is sound and complete for the ground case.

We have not addressed the analogous problem for the general case involving variables. Admitting variables ranging over sets should not present particular problems and, probably, can be incorporated without much difficulty into our framework. However, the significant role is played by the variables wich are allowed to range *only* over individuals (e.g., [Hus93], [WM95b]). Unfortunately, since in general terms denote arbitrary sets, such variables do not admit unrestricted substitution. Consequently, one should not expect the possibility of a straightforward lifting of the presented results to the language containing such variables. This issue is now under investigation.

References

[Bez90] M. Bezem. Completeness of Resolution Revisited. *TCS*, 74, pp.27-237, (1990).

[BG91] L. Bachmair, H. Ganzinger. *Rewrite-Based Equational Theorem Proving with Selection and Simplification.* Technical Report MPI-I-91-208, Max-Planck-Institut f. Informatik, Saarbrücken, (1991).

[BG93] L. Bachmair, H. Ganzinger. *Rewrite Techniques for Transitive Relations.* Technical Report MPI-I-93-249, Max-Planck-Institut f. Informatik, Saarbrücken, (1993). [to appear in LICS'94]

[DJ90] N. Dershowitz, J.-P. Jouannaud. Rewrite systems. In: J. van Leeuwen (ed.) *Handbook of theoretical computer science*, vol. B, chap. 6, pp.243-320. Amsterdam: Elsevier, (1990).

[DM79] N. Dershowitz, Z. Manna. Proving termination with multiset orderings. *Communications of the ACM*, 22:8,pp.465-476, (1979).

[DO92] A. Dovier,E. Omodeo,E. Pontelli,G.-F. Rossi. Embedding finite sets in a logic programming language. *LNAI*, 660, pp.150-167, Springer Verlag, (1993).

[Hes88] W.H. Hesselink. A Mathematical Approach to Nondeterminism in Data Types. *ACM ToPLaS* 10, pp.87-117, (1988).

[Hus92] H. Hussmann. Nondeterministic algebraic specifications and nonconfluent term rewriting. *Journal of Logic Programming*, 12, pp.237-235, (1992).

[Hus93] H. Hussmann. *Nondeterminism in Algebraic Specifications and Algebraic Programs.* Birkhäuser Boston, (1993).

[Jay92] B. Jayaraman. Implementation of Subset-Equational Programs. *Journal of Logic Programming*, 12:4, pp.299-324, (1992).

[Kap88] S. Kaplan. Rewriting with a Nondeterministic Choice Operator. *TCS*, 56:1, pp.37-57, (1988).

[KW94] V. Kriaučiukas, M. Walicki *Reasoning and Rewriting with Set-Relations I: Ground-Completeness.* Technical Report no.96, Dept. of Informatics, University of Bergen (1994).

[LA93] J. Levy, J. Agustí. Bi-rewriting, a term rewriting technique for monotonic order relations. In *RTA'93, LNCS*, 690, pp.17-31. Springer-Verlag, (1993).

[PP91] J. Pais, G.E. Peterson. Using Forcing to Prove Completeness of Resolution and Paramodulation. *Journal of Symbolic Computation*, 11:(1/2), pp.3-19, (1991).

[S-A92] R. Socher-Ambrosius. *Completeness of Resolution and Superposition Calculi.* Technical Report MPI-I-92-224, Max-Planck-Institut f. Informatik, Saarbrücken, (1992).

[SD86] J. Schwartz,R. Dewar,E. Schonberg,E. Dubinsky. *Programming with sets, an introduction to SETL.* Springer Verlag, New York, (1986).

[Sto93] F. Stolzenburg. *An Algorithm for General Set Unification.* Workshop on Logic Programming with Sets, ICLP'93, (1993).

[Wal93] M. Walicki. *Algebraic Specifications of Nondeterminism.* Ph.D. thesis, Institute of Informatics, University of Bergen, (1993).

[WM95] M. Walicki, S. Meldal. A Complete Calculus for Multialgebraic and Functional Semantics of Nondeterminism. [to appear in *ACM ToPLaS*, (1995).]

[WM95b] M. Walicki, S. Meldal. Multialgebras, Power Algebras and Complete Calculi of Identities and Inclusions, *Recent Trends in Data Type Specification, LNCS*, 906, (1995).