

Relative Completeness in Algebraic Specifications

by

Lin Huimin

Relative Completeness in Algebraic
Specifications

LFCS Report Series

ECS-LFCS-87-43

(also published as CSR-252-87)

LFCS

Department of Computer Science
University of Edinburgh
The King's Buildings
Edinburgh EH9 3JZ

December 1987

Copyright © 1987, LFCS

Relative Completeness in Algebraic Specifications

Lin Huimin*

Laboratory for Foundations of Computer Science
Department of Computer Science
University of Edinburgh

December 3, 1987

1 Introduction

A well-known phenomenon in the theory of algebraic specifications is the divergence of different approaches, namely the initial algebra approach [ADJ 78], final algebra approach [Wan 79][Kam 83], and generated model approach [CIP 85]. The notions of behavioral equivalence [GGM 76][ST 86b] and sufficient completeness [GH 78] are closely related to the latter two. Although a lot of work has been done trying to unify these approaches [Rei 81], it seems that more efforts are still needed.

Consistency and sufficient completeness were proposed in [GH 78] in the context of type extension as a requirement to protect base types. [Wan 79][GM 82][Gan 83] proved the existence of the final algebra for a sufficiently complete specification, and it happens to be the 'fully abstract' model in the sense of [Mil 77], i.e. two terms are interpreted as the same element in the model if and only if they are indistinguishable with respect to visible behaviour. [WB 80][CIP 83] showed that the set of all generated models of a sufficiently complete specification forms a complete lattice (w.r.t homomorphism) with the initial and final model being its least and greatest elements, respectively.

*On leave from Institute of Software, Chinese Academy of Sciences, Beijing.

The differences between the initial and final model approaches can be illustrated by the specification of finite sets. Using the initial model semantics, one needs two axioms for the commutative and absorptive laws for the inserting operation. But they are not needed if one works with the final model semantics. It is then argued that the latter is preferable since it has more models which are interesting. But little attention has been paid to the difference between these two specifications within the framework of final model semantics. Both of them are sufficiently complete and their final models are the same, although one of them has two more independent axioms which force its initial model coincide with the final one. These two axioms talk about terms of the new sort, but sufficient completeness concerns only terms in the base sorts. In order to distinguish these two specifications and discuss their relationship, we need a completeness concept which allows us to discuss properties not only in the base sorts but also in the new sort.

In this paper, a concept called *relative completeness* is proposed. Roughly speaking, a relatively complete specification is a maximal specification consistent with the initiality restriction on its base specification. In the context of many-sorted equational theory, it makes sense to talk about relative completeness in different sorts. We call a specification *base complete* if it is relatively complete in each base sort (sort of the base specification), and *complete* if it is relatively complete in every sort. It turns out that base completeness is weaker than sufficient completeness, but it still guarantees the existence of final models.

A complete specification has a unique (up to isomorphism) generated model which protects its base models, so its initial and final models coincide. This is not necessarily true for base complete specifications, whose models are unique only up to behavioural equivalence. But a base complete specification can always be uniquely extended to a complete one and the initial models of the unique extension coincide with the final model of the original specification. Such extension can be characterised as equalizing 'indistinguishable' terms. These results reveal the relationship between the initial and final model semantics.

One may argue that some interesting models would be lost due to such extension and so we introduce another notion: *quotient-initiality*. A model is quotient-initial if it has a quotient model which is isomorphic to the initial model. We show that all (generated) quotient-initial models of a specification form a complete lattice, and the lattice of generated models preserving the base models of a base complete specification can be isomorphically embedded into the lattice of quotient-initial models of its unique complete extension. So, by turning from the initial model to the quotient-initial models, we lose nothing when making complete extensions. Actually we gain more. Some specifications are inherently 'fully abstract'. An

example is the usual specification of stack. The ‘array + pointer’ implementation is not a model of stack specification, but is a quotient-initial model of it.

The paper is organised as follows: Some basic notions and facts of algebraic specifications are summarised in section 2. Section 3 is devoted to the main theoretical development. In section 4 the structures of some interesting sets of models are analyzed. Some examples are given in section 5 to illustrate theoretical results in section 3 and 4. Section 6 generalizes the results of section 3 to handle error-specifications. The relations with previous works are discussed in section 7.

2 Preliminaries

Most of the notations and concepts used in this paper are more or less standard, so we only give a brief introduction here.

Let S be a set of *sorts*. An S -sorted set is a family $X = \{X^s\}_{s \in S}$ of sets indexed by S . If $S' \subseteq S$ then $X^{S'} = \{X^s\}_{s \in S'}$ is an S' -indexed set. We will often omit superscripts and write $x \in X$ rather than $x \in X^s$ for some $s \in S$.

A *signature* is a pair (S, Σ) where S is the sort set and Σ is a family of sets $\{\Sigma^{w,s}\}_{w \in S^*, s \in S}$ which gives the function symbols for the signature. We usually simply write Σ for a signature, with the understanding that the sort set for Σ is S , for Σ' is S' , etc..

Given a signature (S, Σ) and an S -sorted set of variables X , the S -sorted set of Σ -terms over X is denoted by $T_{\Sigma(X)}$. We write T_{Σ} for $T_{\Sigma(\emptyset)}$, the set of all ground Σ -terms. A *ground instance* of a Σ -term t is a ground term obtained by substituting all variables in t with some ground terms, and similar for ground instances of equations.

A Σ -equation is a triple (X, t, t') , where X is a finite set of variables and $t, t' \in T_{\Sigma(X)}$. We usually write $\forall X t = t'$ rather than (X, t, t') .

If E is a set of Σ -equations and e a Σ -equation, $E \vdash e$ denotes that e is derivable from E using first-order equational reasoning (see, for example, [GM 81]).

Lemma 2.1 *Let E be a set of Σ -equations and $t, t' \in T_{\Sigma}$. $E \vdash t = t'$ iff there exist $t_0, \dots, t_n \in T_{\Sigma}$ such that $t = t_0, t' = t_n, t_i = v_i[x/u_i], t_{i+1} = v_i[x/u'_i]$ where $v_i(x) \in T_{\Sigma(X)}, u_i = u'_i$ or $u'_i = u_i$ is a ground instance of some equation in E , $0 \leq i < n$.* $\square$

Let Σ be a signature. A Σ -algebra or Σ -model $A = (D, \varphi)$ consists of an S -sorted set D and an interpretation map φ which assigns each $f : s_1, \dots, s_n \rightarrow s \in \Sigma$ a function $\varphi(f) : D^{s_1} \times \dots \times D^{s_n} \rightarrow D^s$. We usually write $|A|$ for D and f^A for $\varphi(f)$. We also confuse A and $|A|$, and write, for example, $d \in A$ instead of $d \in |A|$.

A Σ -homomorphism h from a Σ -model A to a Σ -model B is a mapping from $|A|$ to $|B|$ such that for any $f : s_1, \dots, s_n \rightarrow s \in \Sigma$ and $a_1 \in |A|^{s_1}, \dots, a_n \in |A|^{s_n}$,

$$h^s(f^A(a_1, \dots, a_n)) = f^B(h^{s_1}(a_1), \dots, h^{s_n}(a_n))$$

A homomorphism $h : A \rightarrow B$ is called an *isomorphism*, denoted $A \cong B$, if there exists a homomorphism $g : B \rightarrow A$ such that $g \circ h = id_A$ and $h \circ g = id_B$, where id_A and id_B are the identity on A and B , respectively.

It is well-known that there exists a unique Σ -homomorphism from T_Σ (with the standard assignment of functions to function symbols in Σ) to any Σ -model A . We call A a *generated-model* if this homomorphism is surjective.

Let $\Sigma \subseteq \Sigma'$. A Σ -model A is called a Σ -submodel of a Σ' -model A' if $|A| \subseteq |A'|^S$ and for every $f : s_1 \dots s_n \rightarrow s \in \Sigma$, $f^A = f^{A'} \cap |M|^{s_1} \times \dots \times |M|^{s_n} \times |M|^s$.

A *congruence* relation $\sim$ on a Σ -model A is an equivalence relation on $|A|$ which is compatible with the operations of A , i.e., $a \sim b_i \Rightarrow f^A(a_1, \dots, a_n) \sim f^A(b_1, \dots, b_n)$ for any $f : s_1 \dots s_n \rightarrow s \in \Sigma$ and $a_i, b_i \in |A|^{s_i}, i = 1, \dots, n$.

Given a congruence $\sim$ on a Σ -model A , the *quotient* model of A by $\sim$ is denoted $A/\sim$. Two Σ -models A and B are said to be *quotient-isomorphic* if there exist congruences $\sim_A$ on A and $\sim_B$ on B such that $A/\sim_A \cong B/\sim_B$. Note that if there exists a surjective homomorphism from A to B then A and B are quotient-isomorphic, but not vice-versa.

Let A be a Σ -model. Let also X be an S -sorted set of variables and v a *valuation* map from X to $|A|$. v can be extended uniquely to a map $v^\sharp : T_{\Sigma(X)} \rightarrow |A|$. For a ground term t , we write t^A for the unique value of t under any valuation.

A Σ -equation $t = t'$ is *satisfied* by a Σ -model A if $v^\sharp(t) = v^\sharp(t')$ for any valuation v . A Σ -model satisfies a set of Σ -equations if it satisfies every equation in the set. We write $A \models E$ when A satisfies E and call A a *model* of E .

A (presentation of a) *specification* is a pair $SP = (\Sigma, E)$ where Σ is a signature and E is a set of Σ -equations called the *axioms* of SP . The class of all models of SP forms a category under Σ -homomorphisms, and $T_{\Sigma, E}$, the quotient model of T_Σ by the congruence on $|T_\Sigma|$ generated by E , is initial in this category.

A *quotient-initial* model of SP is a generated model which has a quotient model

isomorphic to the initial model of SP . It is obvious that the initial model is quotient-initial and any two quotient-initial models are quotient-isomorphic. But note that a quotient-initial model of SP need not be a model of SP , i.e. the equations in E need not be satisfied by it.

Lemma 2.2 *Let I_{SP} be an initial model of $SP = (\Sigma, E)$. For any ground Σ -equation e , $I_{SP} \models e$ iff $E \vdash e$. $\square$*

Notation We use Mod_{SP} to denote the class of all models of SP . GM_{SP} for the set of generated models of SP , and QI_{SP} for the set of quotient-initial models of SP .

Definition 2.1 *Let $SP_1 = (\Sigma, E_1)$ and $SP_2 = (\Sigma, E_2)$ be specifications over the same signature Σ . SP_1 and SP_2 are spec-equivalent if for any $t, t' \in T_\Sigma$, $E_1 \vdash t = t' \iff E_2 \vdash t = t'$.*

Spec-equivalent specifications have the same initial model. This reflects the fact that, semantically, we are only interested in the initial models of basic specifications. As a consequence, for instance, the usual specifications of natural numbers with or without commutative law are spec-equivalent.

3 Extended specifications and relative-completeness

The initial model approach works very well for simple data types such as boolean and integer. But we can hardly specify a real-world system by a single set of equations. We need a way to build up a large specification piece by piece, perhaps in a hierarchical manner, such that specifications at a higher level are extensions of the specifications at lower levels. This leads to the notion of *data type extension*.

In this section we are going to investigate some interesting phenomenon such as *behavioral equivalence* (of models) and *indistinguishability* (between elements of a model), and discuss their relationship with the completeness of specifications. It turns out that the final model, when it exists, is just the initial model of a suitable extension of the original specification.

Definition 3.1 *An extended signature is a pair (Σ_b, Σ) of signatures such that $\Sigma_b \subseteq \Sigma$. Call Σ_b the base signature of the extended signature.*

An extended specification $SP_e = (SP_b, (\Sigma, E_e))$ consists of a base specification $SP_b = (\Sigma_b, E_b)$ and an extension part (Σ, E_e) such that (Σ_b, Σ) is an extended signature and E_e is a set of Σ -equations.

Notation We write $E = E_b \cup E_e$ for the set of axioms of SP_e . For each extended specification SP_e we associate it with a basic specification (Σ, E) . When we talk about ‘the initial model of SP'_e ’, we mean the the initial model of its associated basic specification.

For extended specifications, we are only interested in those models which preserve the initial models of its base specification. This is reflected in the following

Definition 3.2 A model for an extended signature $\Sigma_e = (\Sigma_b, \Sigma)$ is a pair $M_e = (M_b, M)$ such that M is a Σ -model and M_b is a Σ_b submodel of M . Call M_b the base submodel of M . M_e is a generated model for Σ_e if M is a generated model for Σ .

An extended model M of an extended specification $SP_e = (SP_b, (\Sigma, E_e))$ is a generated model (M_b, M) for (Σ_b, Σ) such that M_b is an initial model of SP_b and M satisfies $E = E_b \cup E_e$.

An extended specification is ext-satisfiable if it has an extended model.

Notation The set of extended generated models of SP_e is denoted by $GM_{SP_e}^e$.

We usually write M for an extended model, with the understanding that it has a base submodel M_b .

Although any specification is satisfiable because it always has a model (the singleton model satisfies any set of equations), not every extended specification is ext-satisfiable due to the possible violation of the initiality constraint on its base specification. The following is a simple syntactic characterisation of ext-satisfiability:

Definition 3.3 An extended specification $SP_e = (SP_b, (\Sigma, E_e))$ is consistent iff for any $t, t' \in T_{\Sigma_b}$, $E \vdash t = t' \implies E_b \vdash t = t'$.

By ‘ E is consistent’, where E is a set of Σ -equations, we mean $(SP_b, (\Sigma, E'))$ (as an extended specification) is consistent.

Remark If we consider that the base specification SP_b has as its axioms not just the equations in E_b but also the negations of all those ground Σ_b -equations which

are not provable from E_b ('no confusion' requirement implied in the initiality constraint), the 'real' axioms of SP_b would be

$$E_b \cup \{\neg t = t' : t, t' \in T_{\Sigma_b}, E_b \not\models t = t'\},$$

then *consistency* simply means SP_e is a *conservative extension* of SP_b on Σ_b .

Theorem 3.1 *An extended specification is ext-satisfiable iff it is consistent.*

Proof ($\Rightarrow$) Assume SP_e is ext-satisfiable. Then it has an extended model M with a base-submodel $M_b \cong T_{\Sigma_b, E}$. For any $t, t' \in |T_{\Sigma_b}|$,

$$\begin{aligned} & E_e \vdash t = t' \\ \Rightarrow & M \models t = t' \\ \Rightarrow & M_b \models t = t' \\ \Rightarrow & T_{\Sigma_b, E} \models t = t' \\ \Rightarrow & E_b \vdash t = t'. \end{aligned}$$

Hence SP_e is consistent.

($\Leftarrow$) Assume SP_e is consistent. Let $D_b = \{[t] \in |T_{\Sigma, E}| : \text{there exists } t' \in T_{\Sigma_b}, t' \in [t]\}$. For any $f \in \Sigma_b$, interpret f in D_b as the restriction of $f^{T_{\Sigma, E}}$ on D_b . This turns D_b into a Σ_b model M_b . Define $h : D_b \rightarrow |T_{\Sigma_b, E_b}|$ by $h([t]) = [t]_b$ for $t \in |T_{\Sigma_b}|$, where $[t]_b \in |T_{\Sigma_b, E_b}|$ is the equivalence class of t in T_{Σ_b, E_b} . Then the consistency of SP_e guarantees $h : M_b \cong T_{\Sigma_b, E_b}$. By construction, M_b is a base-submodel of $T_{\Sigma, E}$. Hence SP_e has an extended model $T_{\Sigma, E}$. $\square$

Corollary 3.2 *An extended specification is consistent iff its initial model is an extended model.*

For extended specifications, it makes sense to think that only the base sorts are 'visible' while the new sorts are 'hidden', i.e., we do not want to distinguish between two extended models so far as they have the same properties in base sorts, although their internal structures may be different [GGM 76][Gan 83][Kam 83][Rei 81][Sch 83][ST 86b].

Definition 3.4 *Let SP_e be an extended specification. Two extended models M, M' of SP_e are s-equivalent (where s is a sort), iff for any $t, t' \in T_{\Sigma}^s$, $M \models t = t' \iff M' \models t = t'$.*

Two extended models are behaviourally equivalent, or equivalent, if they are s-equivalent for every $s \in S_b$, or for every $s \in S$, respectively.

The syntactic characterisation of the above equivalence notion between extended models turns out to be some kind notion of completeness.

Definition 3.5 *An extended specification $SP_e = (SP_b, (\Sigma, E_e))$ is s -relative-complete ($s \in S$ is a sort), if for any $t, t' \in T_{\Sigma}^s$, $\{t = t'\} \cup E$ is consistent $\implies E \vdash t = t'$.*

An extended specification is base-relative-complete, or relative-complete, if it is s -relative-complete for every $s \in S_b$, or for every $s \in S$, respectively.

We will drop the adjective ‘relative’ in the rest of this paper for brevity.

Remark There are three equivalent definitions of completeness of a first-order theory: every sentence which is consistent with the theory is provable from the theory; all models of the theory are elementary equivalent, i.e. the same set of sentences hold in all its models; or, for each sentence, either itself or its negation is provable from the theory. The last one makes no sense in our system since we do not (explicitly) have negation. But we do have the notion of consistency, which is relative to the initiality constraint on the base-specifications, so we can define completeness by means of consistency. Note the similarity between our notion of equivalence and *elementary equivalence*. In fact, we have a similar result connecting the syntactic notion of completeness and the semantic notion of equivalence.

Theorem 3.3 *A consistent extended specification SP_e is s -complete iff all its extended models are s -equivalent.*

Proof $(\implies)$ Assume SP_e is s -complete. Let M, M' be its extended models. For any $t, t' \in T_{\Sigma}^s$,

$$\begin{aligned} & M \models t = t' \\ \implies & M \text{ is an extended model of } E \cup \{t = t'\} \\ \implies & E \cup \{t = t'\} \text{ is consistent} & (\text{Theorem 3.1}) \\ \implies & E \vdash t = t' \\ \implies & M' \models t = t' \end{aligned}$$

By the same argument, $M' \models t = t' \implies M \models t = t'$.

$(\impliedby)$ Assume all extended models of SP_e are s -equivalent. By Corollary 3.2, $T_{\Sigma, E}$ is an extended model of SP_e . For any $t, t' \in T_{\Sigma}^s$,

$$E \cup \{t = t'\} \text{ is consistent}$$

$\Rightarrow E \cup \{t = t'\}$ has an extended model M (Theorem 3.1)
 $\Rightarrow M$ is an extended model of SP_e and $M \models t = t'$
 $\Rightarrow T_{\Sigma, E} \models t = t'$
 $\Rightarrow E \vdash t = t'$

Hence SP_e is s-complete. □

Corollary 3.4 1. A consistent specification is base-complete iff all its extended models are behaviorally equivalent.

2. A consistent specification is complete iff all its extended models are equivalent.

A stronger result holds for complete specifications:

Theorem 3.5 A consistent specification is complete iff its extended models are initial.

Proof ($\Rightarrow$) Assume SP_e is complete. Let M be its extended model. M is equivalent to $T_{\Sigma, E}$ by the above corollary. Define a map $h : T_{\Sigma, E} \rightarrow M$ by $h(t^{T_{\Sigma, E}}) = t^M$. It's easy to check that h is well-defined and is in fact an isomorphism.

($\Leftarrow$) Isomorphic models are equivalent, so completeness follows from the above corollary. □

Before going into the discussion on the relationship between base-completeness and completeness, let's have a look at an example of complete specifications.

```

POS_INT = sort int
  opns 0 :→ int
        S : int → int
        + : int int → int
  eqns x + 0 = x
        x + S(y) = S(x + y)
end

```

```

INT = base POS_INT
  opns P : int → int
        - : int int → int
  eqns P(S(x)) = x

```

$$\begin{aligned}
&S(P(x)) = x \\
&x + P(y) = P(x + y) \\
&x - 0 = x \\
&x - S(y) = P(x - y) \\
&x - P(y) = S(x - y)
\end{aligned}$$

end

It is easy to show that INT is complete. Since, up to renaming, POS_INT is the same as the usual specification of natural numbers, say NAT, INT can be seen as an complete extension of NAT. The above theorem says that so long as the standard model of natural numbers is protected, INT has a unique (generated) model, which is the standard model of integers.

An extended specification subjects to two different kinds of restrictions: the initiality constraint (because we want to have standard model) and the extension constraint (because we want to protect the base specification). Theorem 3.5 tells us that these two restrictions coincide for complete specifications. One consequence of this is that we can then reduce complete extended specifications to simple ones : an extended specification $SP_e = (SP_b, (\Sigma, E_e))$ is semantically the same as the basic specification $(\Sigma, E_b \cup E_e)$. On the other hand, a basic specification can be viewed as an extended one: $SP = (SP, (\Sigma, \emptyset))$ (i.e., extended with nothing), then it is trivially consistent and complete. So if we insist on completeness, the semantics of algebraic specifications could be simplified greatly. But requiring completeness in every sort seems asking too much, in view of the fact that we are only interest in the visible behaviour of abstract data types. It seems that base-completeness would be adequate, as suggested by corollary 3.4 (1). So a question rises: Is there any inherent relation between base-completeness and completeness? In order to answer this question, we need the following concept:

Definition 3.6 *A specification $SP_e = (SP_b, (\Sigma, E_e))$ is maximally consistent iff it is consistent and for any $t, t' \in T_\Sigma, \{t = t'\} \cup E$ is consistent $\implies t = t' \in E$*

By definition, a maximal consistent specification is complete.

Given any consistent specification SP_e , we can extend it to a maximal consistent one as follows:

Enumerate all ground Σ -equations:

$$e_1, e_2, \dots, e_i, \dots$$

Define $E_i, i = 0, 1, \dots$, by

$$\begin{aligned} E_0 &= E \\ E_i &= \begin{cases} \{e_i\} \cup E_{i-1} & \text{if } \{e_i\} \cup E_{i-1} \text{ consistent} \\ E_{i-1} & \text{otherwise} \end{cases} \end{aligned}$$

Let $E^* = \bigcup_i E_i$. By construction, E^* is maximally consistent.

If we change the order of e_i 's in the above construction, we may end up with a different E^* . In general, a consistent specification may have several (even infinitely many) different maximal consistent extensions. But for base-complete specifications, we have

Theorem 3.6 *A consistent and base-complete specification has a unique maximal consistent extension (up to spec-equivalence).*

To prove this theorem, we need a lemma.

Lemma 3.7 *Suppose SP_e is consistent and base-complete. For any ground Σ -equations e, e' , if both $\{e\} \cup E$ and $\{e'\} \cup E$ are consistent then $\{e, e'\} \cup E$ is also consistent.*

Proof Suppose $\{e, e'\} \cup E \vdash t = t'$ for arbitrary $t, t' \in T_{\Sigma_b}$. By lemma 2.1, there exist $t_0, t_1, \dots, t_n \in T_{\Sigma}^{S_b}$ such that $t_0 = t, t_n = t', t_i = v_i[x/u_i], t_{i+1} = v_i'[x/u_i'], v_i(x) \in T_{\Sigma(X)}, u_i = u_i'$ or $u_i' = u_i$ is a ground instance of some equation e_i in $\{e, e'\} \cup E, 0 \leq i < n$.

If $e_i \in E$, then $E \vdash t_i = t_{i+1}$. Otherwise $u_i = u_i'$ or $u_i' = u_i$ must be one of e and e' , so $\{e\} \cup E \vdash t_i = t_{i+1}$ or $\{e'\} \cup E \vdash t_i = t_{i+1}$. Since both $\{e\} \cup E$ and $\{e'\} \cup E$ are consistent, so is $\{t_i = t_{i+1}\} \cup E$. Thus $E \vdash t_i = t_{i+1}$ by the base-completeness of SP_e . Hence $E \vdash t_i = t_{i+1}$ in either case, and $E \vdash t = t'$ follows. By the consistency of SP_e , $E_b \vdash t = t'$. $\square$

Proof of Theorem 3.6 Let $E^* = E \cup E^o$, where $E^o = \{t = t' : t, t' \in T_{\Sigma} \text{ and } \{t = t'\} \cup E \text{ is consistent}\}$. Suppose E^* is inconsistent. Then there must be a finite subset E' of E^* such that $E \cup E'$ is inconsistent. This is in contrary with the above lemma. Hence E^* must be consistent. It is maximal by the construction.

Now suppose A^o is another maximal consistent extension of $Spec_e$. For any $t, t' \in T_{\Sigma}$, $A^o \vdash t = t' \implies t = t' \cup A^o$ is consistent $\implies t = t' \cup A$ is consistent $\implies t = t' \in A^* \implies A^* \vdash t = t'$. By the same argument, $A^* \vdash t = t' \implies A^o \vdash t = t'$. Hence A^o and A^* are spec-equivalent. $\square$

Notice that, in constructing the maximal consistent extension, we add to the axiom set all ground equations which are consistent with it, and base-completeness guarantees the consistency of the resulted axiom set. Among these ground equations, those in base sorts are already in the inference closure of the original axiom set because it is base-complete. Thus what we essentially added are some equations in new sorts. We would like to give a characterisation for these equations.

A *base context* is a one-variable term of a base sort.

Theorem 3.8 *Suppose SP_e is consistent. The following are equivalent:*

1. SP_e is base-complete;
2. For any $t, t' \in T_\Sigma$, $\{t = t'\} \cup E$ is consistent iff $E \vdash cont(t) = cont(t')$ for every base context $cont(x)$.

Proof (1 $\implies$ 2) Assume SP_e is base-complete.

($\implies$) Assume $\{t = t'\} \cup E$ is consistent. Then $\{cont(t) = cont(t') \cup E\}$ is consistent too. Therefore $E \vdash cont(t) = cont(t')$ by the base-completeness of SP_e .

($\impliedby$) Assume $E \vdash cont(t) = cont(t')$ for every base context $cont(x)$. Suppose $\{t = t'\} \cup E \vdash r = r'$ where $r, r' \in T_{\Sigma_b}$. Then there exist $r_0, r_1, \dots, r_n \in T_\Sigma^{S_b}$ such that $r_0 = r, r_n = r', r_i = v_i[x/u_i], r_{i+1} = v_i[x/u'_i], v_i(x) \in T_{\Sigma(X)}$, either $u_i = u'_i$ or $u'_i = u_i$ is a ground instance of some equation $e \in \{t = t'\} \cup E, 0 \leq i < n$.

If $e \in E$, then it is obvious that $E \vdash r_i = r_{i+1}$. Otherwise e must be $t = t'$. Note that v_i is a base context, so $E \vdash v_i(t) = v_i(t')$ by the hypothesis. Hence $E \vdash r_i = r'_i$ in either case, and $E \vdash r = r'$ follows. Since SP_e is consistent and $r = r' \in T_{\Sigma_b}$, $E_b \vdash r = r'$. This shows $\{t = t'\} \cup E$ is consistent.

(2 $\implies$ 1) Assume that $\{t = t'\} \cup E$ is consistent iff $E \vdash cont(t) = cont(t')$ for arbitrary base context $cont(x)$. Take a particular base context $x \in X_{S_b}$, we have $\{t = t'\} \cup E$ is consistent $\implies E \vdash t = t'$ for any $t, t' \in T_\Sigma^{S_b}$, i.e. SP_e is base-complete. $\square$

Two terms satisfying the condition of this theorem are usually called *indistinguishable* in the literature, as they have the same observable properties. What we did in extending a base-complete specification to maximal consistent one was to equalize all pairs of indistinguishable terms. As the result, the initial model of the unique maximal extension is the final model of the original specification

in the sense of the final model approach. In fact, there is an implicit understanding behind the specification methods based on the notion of ‘behavioural equivalence’: indistinguishable terms are considered equal. In general, equalizing different terms might cause inconsistency, but for base-complete specifications this will not happen, which was made clear by Theorems 3.6 and 3.8. The construction of maximal consistent extension actually ‘realizes’ this meta-level assumption at object language level.

4 Embedding $GM_{SP_e}^e$ into $QI_{SP_e}^e$

In making the maximal consistent extension of a base-complete specification, some interesting models might be lost. We want to get them back in some way. It is to this end we introduced the notion of *quotient-initial* model.

In this section, we are going to investigate the structures of extended models and quotient-initial models of an extended specification, as well as the relationship between them.

Let Σ be a signature. All Σ -congruence relations on T_Σ forms a complete lattice $(C_\Sigma, \subseteq)$ under set inclusion. For an arbitrary $\sim \in C_\Sigma$, $C_\Sigma \downarrow \sim = \{c \in C_\Sigma : c \subseteq \sim\}$, the set of all *congruences under* $\sim$, forms a complete sublattice of C_Σ . This is also true for $C_\Sigma \uparrow \sim = \{c \in C_\Sigma : \sim \subseteq c\}$, the set of all *congruences over* $\sim$.

A Σ -model M induces a congruence relation $\sim_M$ on T_Σ given by $t \sim_M t'$ iff $M \models t = t'$ for any $t, t' \in |T_\Sigma|$. If M is a generated model, then $M \cong T_\Sigma / \sim_M$.

Let M, M' be Σ -models. We write $M \prec M'$ if there exists a Σ -homomorphism from M to M' . It is well-known that $M \prec M'$ iff $\sim_M \subseteq \sim_{M'}$ for generated models M, M' . So we can ‘transfer’ the lattice structure of C_Σ to GM_Σ and obtain a complete lattice $(GM_\Sigma / \cong, \prec)$.

Let $SP = (\Sigma, E)$ be a basic specification and M a quotient-initial model of SP . $M \prec T_{\Sigma, E}$ by definition, so $\sim_M \subseteq \equiv_E$, i.e. $\sim_M \in C_\Sigma \downarrow \equiv_E$. On the other hand, suppose $\sim \in C_\Sigma \downarrow \equiv_E$, we can define a map $h : T_\Sigma / \sim \rightarrow T_\Sigma / \equiv_E$ by $h([t]_\sim) = [t]_{\equiv_E}$ for $t \in T_\Sigma$. It is easy to verify that h is well-defined and in fact is a Σ -homomorphism. This shows that $T_\Sigma / \sim$ is a quotient-initial model of SP .

In this way, we obtain a bijection between $(QI_{SP} / \cong, \prec)$ and $(C_\Sigma \downarrow \equiv_E, \subseteq)$. It is easy to see that this bijection preserves ordering. We already know $(C_\Sigma \downarrow \equiv_E, \subseteq)$ is a complete lattice, so this gives

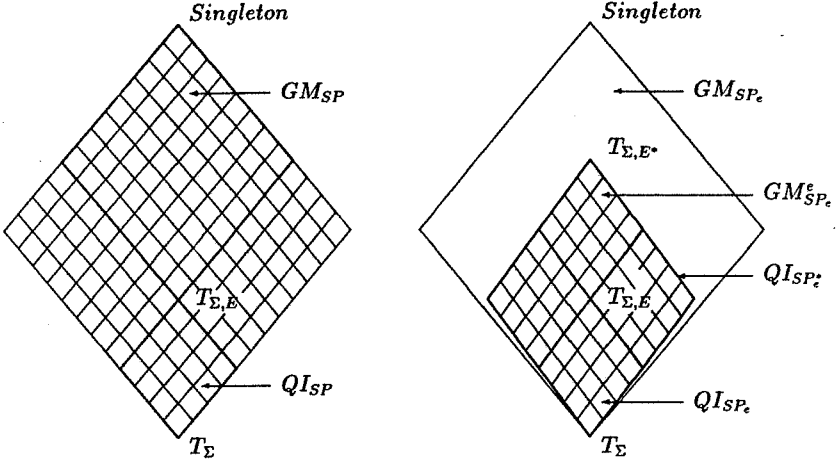


Figure 1:

Fact 4.1 *The set of all quotient-initial models of a specification forms a complete lattice under homomorphism (modulo isomorphism), with its initial model being the greatest element (i.e. the initial model is final in the category of quotient-initial models and homomorphisms).*

A fact from universal algebra says that $(GM_{SP}/\cong, \prec)$ is isomorphic to $(C_{\Sigma} \upharpoonright \equiv_E, \subseteq)$, so the former is also a complete lattice. The situation is pictured in Figure 1 (left).

In the context of extended specifications, we are more interested in extended models than simple ones, so we extend the above notions accordingly.

In the following discussion we fix a consistent extended specification $SP_e = (SP_b, (\Sigma, E_e))$. We write $\equiv_b$ for the congruence relation on T_{Σ_b} induced by E_b and $\equiv$ for the congruence relation on T_{Σ} induced by $E = E_e \cup E_b$. An extended congruence $\sim$ is a congruence on T_{Σ} such that $\sim|_{\Sigma_b} = \equiv_b$. Let C_{SP_e} denote the set of all extended congruences on T_{Σ} .

Fact 4.2 1. $M \in |GM_{SP_e}^i|$ iff $\sim_M \in C_{SP_e}$.

2. SP_e is consistent iff $\equiv \in C_{SP_e}$.

3. $(C_{SP_e}, \subseteq)$ is a lower semilattice with the least element $\equiv$.

Define a binary relation $\bowtie$ on CS_{P_e} by: $c \bowtie c'$ iff $c|_{T_\Sigma|S_b} = c'|_{T_\Sigma|S_b}$. $\bowtie$ is an equivalence relation. It is easy to see that $c \bowtie c'$ if and only if T_Σ/c and T_Σ/c' are behaviorally equivalent.

Suppose SP_e is base-complete, then $c \bowtie c'$ for any $c, c' \in CS_{P_e}$, which is an obvious consequence of corollary 3.4(1). SP_e has a unique maximal extension SP_e^* with axiom set $E^* \equiv_{E^*} \in CS_{P_e}$ by fact 4.2(1). Furthermore, for any $\sim \in CS_{P_e}, t \sim t' \implies \{t = t'\} \cup E$ is consistent $\implies t = t' \in E^* \implies t \equiv_{E^*} t'$, i.e. $\sim \prec \equiv_{E^*}$. In other words, $\equiv_{E^*}$ is the maximal element of CS_{P_e} . Combined with fact 4.2(3), this shows that $(CS_{P_e}, \subseteq)$ is a complete lattice. Rephrased in the language of models and homomorphisms, this gives

Theorem 4.3 *Suppose SP_e is consistent and base-complete. $GM_{SP_e}^e / \cong$ forms a complete lattice under Σ -homomorphism.*

When SP_e is complete, $GM_{SP_e}^e / \cong$ degenerates to a singleton lattice.

An *ext-quotient-initial model* of SP_e is a generated model for (Σ_b, Σ) that is quotient-initial and its base submodel is an initial model of SP_b . We denote the set of all ext-quotient-initial model of SP_e by $QI_{SP_e}^e$.

Let $C_\Sigma^e \downarrow \equiv \{c \in C_\Sigma \downarrow : c|_{\Sigma_b} \equiv_b\}$. $\equiv \in C_\Sigma^e \downarrow \equiv$ and is the greatest element of $(C_\Sigma^e \downarrow \equiv, \subseteq)$. For an arbitrary non-empty subset N of $C_\Sigma^e \downarrow \equiv$, $c_N = \bigcap N$ is the greatest lower bound of N in $(C_\Sigma \downarrow \equiv, \subseteq)$. Since for any $c \in N$, $c|_{\Sigma_b} \equiv_b$, we have $c_N|_{\Sigma_b} \equiv_b$, i.e. $c_N \in C^e \downarrow \equiv$ and is the greatest lower bound of N in $(C_\Sigma^e \downarrow \equiv, \subseteq)$. This shows $(C_\Sigma^e \downarrow \equiv, \subseteq)$ is a complete sublattice of $(C_\Sigma \downarrow \equiv, \subseteq)$. In much the same way as we did in the proof of fact 4.1, we can define a bijection from $QI_{SP} / \cong$ to $C_\Sigma \downarrow \equiv$ which preserves the respective orders. This gives the following

Fact 4.4 *The set of all ext-quotient-initial models of a consistent specification forms a complete lattice under homomorphism (modulo isomorphism), the initial model being the greatest element (i.e. the initial model is final in the category of ext-quotient-initial models and homomorphisms).*

We have seen that, for a base-complete specification, both its extended models and ext-quotient-initial models form complete lattices under homomorphism. The only point they intersect at is the initial model, which is final among ext-quotient-initial models. This seems uninteresting. But what about the ext-quotient-initial models of its unique maximal consistent extension?

Let SP_e be a base-complete specification and $SP_e^* = (\Sigma, E^*)$ its unique maximal consistent extension. Let also M be an extended model of SP_e . Define a binary

relation $\sim_M^*$ on $|M|$ by : for $d, d' \in |M|$, $d \sim_M^* d'$ iff there exist $t, t' \in T_E$ such that $d = t^M, d' = t'^M$ and t, t' are indistinguishable. From Theorems 3.6, 3.8 and the construction of maximal consistent extension, $M / \sim_M^* \cong T_{E, E^*}$. In this way we obtain a map $\tau : GM_{SP_e}^E \rightarrow QISP_e$ which is injective (modulo isomorphism). It is easy to verify that $\tau(M) \prec \tau(M')$ iff $M \prec M'$. So we have the following

Theorem 4.5 ($GM_{SP_e}^E, \prec$) can be isomorphically embedded into $(QISP_e, \prec)$.

We depict this fact in Figure 1 (right).

5 Examples

We give three different specifications of finite set of natural numbers. In the following, we assume the standard specification **BOOL** of boolean and **NAT** of natural numbers (with **Eq**, the equality predicate).

```

SET0 = base BOOL, NAT
      sort set
      opns  $\emptyset : \rightarrow set$ 
            Insert, Delete : nat set  $\rightarrow set$ 
             $\in : nat set \rightarrow bool$ 
            If_Then_Else : bool bool bool  $\rightarrow bool$ 
      eqns  $n \in \emptyset = False$ 
             $n \in Insert(m, s) = If Eq(n, m) Then True Else n \in s$ 
             $n \in Delete(m, s) = If Eq(n, m) Then False Else n \in s$ 
            If True Then b1 Else b2 = b1
            If False Then b1 Else b2 = b2
      end

SET1 = base BOOL, NAT
      sort set
      opns  $\emptyset : \rightarrow set$ 
            Insert, Delete : nat set  $\rightarrow set$ 
             $\in : nat set \rightarrow bool$ 
            Ifset_Then_Else : bool set set  $\rightarrow set$ 
            Ifbool_Then_Else : bool bool bool  $\rightarrow bool$ 
      eqns Delete(n,  $\emptyset$ ) =  $\emptyset$ 
            Delete(n, Insert(m, s)) = Ifset Eq(n, m) Then Delete(n, s)
                                     Else Insert(m, Delete(n, s))

```

```

     $n \in \emptyset = \text{False}$ 
     $n \in \text{Insert}(m, s) = \text{If}_{\text{bool}} \text{Eq}(n, m) \text{ Then True Else } n \in s$ 
    (plus the obvious axioms for If_Then_Else)
end

SET2 = base BOOL, NAT
sort set
opns  $\emptyset : \rightarrow \text{set}$ 
       $\text{Insert}, \text{Delete} : \text{nat set} \rightarrow \text{set}$ 
       $\in : \text{nat set} \rightarrow \text{bool}$ 
       $\text{If}_{\text{set}} \text{ _Then_ Else} : \text{bool set set} \rightarrow \text{set}$ 
       $\text{If}_{\text{bool}} \text{ _Then_ Else} : \text{bool bool bool} \rightarrow \text{bool}$ 
eqns  $\text{Delete}(n, \emptyset) = \emptyset$ 
       $\text{Delete}(n, \text{Insert}(m, s)) = \text{If}_{\text{set}} \text{Eq}(n, m) \text{ Then } \text{Delete}(n, s)$ 
       $\text{Else } \text{Insert}(m, \text{Delete}(n, s))$ 
       $\text{Insert}(n, \text{Insert}(m, s)) = \text{Insert}(m, \text{Insert}(n, s))$ 
       $\text{Insert}(n, \text{Insert}(n, s)) = \text{Insert}(n, s)$ 
       $n \in \emptyset = \text{False}$ 
       $n \in \text{Insert}(m, s) = \text{If}_{\text{bool}} \text{Eq}(n, m) \text{ Then True Else } n \in s$ 
      (plus the obvious axioms for If_Then_Else)
end

```

Using induction on the structure of terms, it is not difficult to check that all three specifications above are base-complete (in fact they are sufficiently complete) but only SET2 is complete. Furthermore, the maximal consistent extensions of SET0 and SET1 are equivalent to SET2.

Now consider three models *Set*, *Seq* and *SeqPair*. The interpretations for sorts *bool* and *nat* are standard. In *Set*, the carrier for the sort *set* is just finite sets of natural numbers and the function symbols are interpreted as the usual set operations. *Seq* and *SeqPair* are defined as follows:

Seq:

$|Seq|^{\text{set}}$ = the set of all finite sequences of natural numbers.

$\emptyset^{\text{Seq}}$ = the empty sequence.

$\text{Insert}^{\text{Seq}}(n, s)$ = append n to the right end of s .

$\text{Delete}^{\text{Seq}}(n, s)$ = delete all occurrences of n from s .

$n \in^{\text{Seq}} s$ = if n occurs in s then true else false.

The interpretations of *If Then Else* are as expected.

SeqPair:

$|SeqPair|^{set}$ = the set of all finite sequences of (i', n) and (d', n) where n is a natural number.

$\emptyset^{Seq}$ = the empty sequence.

$Insert^{SeqPair}(n, s)$ = append (i', n) to the right end of s .

$Delete^{SeqPair}(n, s)$ = append (d', n) to the right end of s .

$n \in^{SeqPair} s$ = Search in s from right to left for the first pair whose second component equals n . If there is such a pair and its first component is i' , then the result is true, otherwise the result is false.

Intuitively, all three models above are correct implementations of any reasonable specifications of finite set. It is easy to see that $SeqPair \models SET0$, $Seq \models SET0, SET1$, $Set \models SET0, SET1, SET2$. Also, $SeqPair, Seq, Set$ are the initial models of $SET0, SET1, SET2$, respectively, and Set is the final model of $SET0, SET1$ and $SET2$. Since Seq is not a model of $SET2$ and $SeqPair$ is not a model of $SET1$, it is might argued that these two specifications are too restrictive. They dictate the internal structure on the sort *set* which is supposed to be hidden from users, this seems violating the principle of data abstraction. But if we disallow mentioning any properties on invisible sorts, we would not be able to finitely specify some data types like stack. What's the solution then?

If we examine the above models more carefully, we'll find that they are quotient-isomorphic. In fact, for Seq , let $s1 \sim_{Seq} s2$ iff the same natural numbers occur in sequences $s1$ and $s2$; for $SeqPair$, let $s1 \sim_{SeqPair} s2$ iff for any n , when the first pair with second component equals n in $s1$ has first component i then it must be so in $s2$, and vice versa ($\sim_{Seq}$ and $\sim_{SeqPair}$ are identity on other sorts), we have $Seq / \sim_{Seq} \cong SeqPair / \sim_{SeqPair} \cong Set$. So, taking the maximal consistent extension of the three specifications, the sets of quotient-initial models of these extensions are the same and include all the models we are interested in.

6 Adding Error Constants to Base Sorts

Specifications of some data types will inevitably lead to exceptional situations, which can be modeled by introducing error constants. But then these specifications are no longer sufficiently complete, since error constants can not (and should not) be equal to any term of base specification.

The definitions of relative-completeness and extended models do not exclude new constants of base sorts, so it is possible to handle 'error specifications' within our framework. We give a simple treatment in this section.

Definition 6.1 An extended error-specification is an extended specification $SP_e^{err} = (SP_b, (\Sigma, E_e))$ with a designated subset of Σ : $ERR_\Sigma = \{Err_s : \rightarrow s : s \in S_b\}$. For simplicity, we assume there is at most one such error constant in each base sort.

The axiom set of SP_e^{err} is $E = \bar{E}_b \cup E_e$, where $\bar{E}_b = \{t = t' : t, t' \in T_{\Sigma_b}, E_b \vdash t = t'\}$.

Note that the axioms of the base specification are replaced by their ground inference closure so that the inconsistency which might be caused by ‘error propagation’ axioms can be avoided.

Definition 6.2 An extended error-model of an extended error-specification SP_e^{err} is an extended model M of SP_e^{err} considered as an ordinary extended specification and $Err_s^M \notin |M_b|^s$ for any $Err_s \in ERR_\Sigma$.

Definition 6.3 An extended error-specification SP_e^{err} is consistent if for any $t, t' \in |T_{\Sigma_b}|^s$, $Err_s \in ERR_\Sigma$, $E \vdash t = t' \implies E_b \vdash t = t'$, and $E \not\vdash t = Err_s$.

An extended error-specification is called *ext-satisfiable* if it has an extended error-model.

These definitions reflect our intention that error elements must be distinct from normal elements.

Now parallel to theorem 3.1, we have

Theorem 6.1 An extended error-specification is consistent iff it is ext-satisfiable.
□

The proof is the same as that of theorem 3.1, with additional straightforward verification of the extra condition concerning error constants.

The notions of relative-completeness for error-specifications, of maximal consistency, and of behavioral equivalence of extended error-models are the same as those in section 3. Theorems 3.3, 3.5, 3.6, 3.8 and their corollaries hold for error-specifications and the proofs go through almost without change except for the proof of lemma 3.7 and *if* part of the proof of theorem 3.8, where one more case for error constants needs to be checked and that is trivial.

Here is an example of error-specification which enriches the usual specification of natural numbers with minus operation.

```

NAT = sort nat
      opns  $0 : \rightarrow \text{nat}$ 
             $S : \text{nat} \rightarrow \text{nat}$ 
             $+: \text{nat } \text{nat} \rightarrow \text{nat}$ 
      eqns  $x + 0 = x$                  $x + S(y) = S(x + y)$ 
             $x * 0 = 0$                  $x * S(y) = x + x * y$ 
      end

NAT_WITH_MINUS = base NAT
      opns  $P : \text{nat} \rightarrow \text{nat}$ 
             $- : \text{nat } \text{nat} \rightarrow \text{nat}$ 
             $Err : \rightarrow \text{nat}$ 
      eqns  $P(0) = Err$                  $P(S(x)) = x$ 
             $x - 0 = x$                  $x - S(y) = P(x - y)$ 
             $S(Err) = Err$              $P(Err) = Err$ 
             $x + Err = Err$              $Err + x = Err$ 
             $x - Err = Err$              $Err - x = Err$ 
             $x * Err = Err$              $Err * x = Err$ 
      end

```

Note that, according to definition 6.1, the axiom set of **NAT_WITH_MINUS** includes only the ground inference closure (within the signature of **NAT**) of the axioms of **NAT**, rather than the axioms themselves. So $Err * 0 = 0$ is not provable in the above error-specification.

It is easy to check that **NAT_WITH_MINUS** is consistent and complete.

7 Related work

7.1 Sufficient-completeness and final model semantics

The work reported in this paper is based on the notion of ‘relative-completeness’. A closely related notion is *sufficient-completeness* which is well-known and can be stated as follows:

Let $SP_e = (SP_b, (\Sigma, E_e))$ be an extended specification. SP_e is sufficiently-complete if for any $t \in |T_\Sigma|^{S_b}$, there exists $\hat{t} \in |T_\Sigma|$ such that $E \vdash t = \hat{t}$.

Theorem 7.1 *Sufficient-completeness is strictly stronger than base-completeness.*

Proof Assume SP_e is consistent and sufficiently-complete. For arbitrary $t, t' \in |T_\Sigma|^{S_b}$, suppose $\{t = t'\} \cup E$ is consistent. Then there exist $\hat{t}, \hat{t}' \in T_{\Sigma_b}$ such that

$$E \vdash t = \hat{t}, \quad E \vdash t' = \hat{t}'. \quad (*)$$

From these we can deduce $\{t = t'\} \cup E \vdash \hat{t} = \hat{t}'$. But $\hat{t}, \hat{t}' \in T_{\Sigma_b}$, so $E_b \vdash \hat{t} = \hat{t}'$ because SP_e is consistent. Since $E_b \subseteq E$, $E \vdash \hat{t} = \hat{t}'$. Combined with (*), this gives $E \vdash t = t'$, i.e. SP_e is base-complete.

On the other hand, there are base-complete specifications which are not sufficiently-complete. Here is a simple example:

```

TWO = sort s
      opns a, b :→ s
      end

THREE_FROM_TWO = base TWO
                  opns c :→ s
                  f : s → s
                  eqns f(a) = b
                      f(b) = c
                      f(c) = a
                  end

```

It is easy to show that THREE_FROM_TWO is base-complete (also complete in this case since there is no new sort) by induction. But THREE_FROM_TWO is not sufficiently-complete, because, for instance, the ground term c can not be reduced to either a or b which are the only ground terms in the base signature. $\square$

Since sufficient-completeness is a special case of base-completeness and our notion of 'extended model' coincides with 'hierarchical model' [CIP 83] when specifications are sufficiently-complete, many of our results in section 3 can be seen as generalizations of similar works based on sufficient-completeness.

7.2 Correspondence and quotient-isomorphism

[Sch 83] proposed the notion of (*strong*-)correspondence between two algebras as a characterization of behavioural equivalence. For two Σ - algebra A and B (we

only consider total algebra here), a binary relation $G \subseteq |A| \times |B|$ is called a correspondence from A to B if it is compatible with the algebraic structures of A and B , i.e.

$$(a_i, b_i) \in G \implies (f^A(a_1, \dots, a_n), f^B(b_1, \dots, b_n)) \in G,$$

for any $a_i \in |A|^{s_i}, b_i \in |B|^{s_i}, i = 1, \dots, n$, and every $f : s_1 \dots s_n \rightarrow s \in \Sigma$.

A correspondence G from A to B is called *full* if for any $a \in |A|$, there is a $b \in |B|$ such that $(a, b) \in G$ and for any $b \in |B|$, there is an $a \in |A|$ such that $(a, b) \in G$. It turns out that full correspondence is equivalent to quotient-isomorphism.

Fact 7.2 *Let A, B be two Σ -algebras. There exists a full correspondence from A to B iff A and B are quotient-isomorphic.*

Proof If A and B are quotient-isomorphic, i.e. there are congruences $\sim_A$ on A and $\sim_B$ on B such that $h : A/\sim_A \cong B/\sim_B$, then $G = (a, b) : h([a]_{\sim_A}) = [b]_{\sim_B}$, where $[a]_{\sim_A}$ is the equivalent class of a in A w.r.t $\sim_A$ and $[b]_{\sim_B}$ is the equivalent class of b in B w.r.t $\sim_B$, is clearly a full correspondence from A to B .

Conversely, suppose G is a full correspondence from A to B . Define two sequences of binary relations $\sim_i^{GA}$ and $\sim_i^{GB}$ on $|A|$ and $|B|$, respectively, as follows:

$$\begin{aligned} \sim_0^{GA} &= id_{|A|} & \sim_0^{GB} &= id_{|B|} \\ \sim_i^{GA} &= G \otimes \sim_{i-1}^{GB} \otimes G^{op} & \sim_i^{GB} &= G^{op} \otimes \sim_{i-1}^{GA} \otimes G \end{aligned}$$

where $G^{op} = \{(b, a) : (a, b) \in G\}$, and for $U \subseteq M \times N, V \subseteq N \times P$ and $W \subseteq P \times Q$, $U \otimes V \otimes W = \{(m, q) \in M \times Q : \text{there exist } n \in N \text{ and } p \in P \text{ such that } (m, n) \in U, (n, p) \in V \text{ and } (p, q) \in W\}$.

From this definition, we have $\sim_i^{GA} \subseteq \sim_{i+1}^{GA}$ and $\sim_i^{GB} \subseteq \sim_{i+1}^{GB}$ because G is full.

Let $\sim^{GA} = \bigcup_i \sim_i^{GA}$ and $\sim^{GB} = \bigcup_i \sim_i^{GB}$. We claim that $\sim^{GA}$ and $\sim^{GB}$ are congruence relations on A and B , respectively. It is straightforward to verify that they are equivalence relations. To see that they are compatible with the algebraic structure, we apply induction on i to show that, for each i , $\sim_i^{GA}$ and $\sim_i^{GB}$ are substitutable w.r.t. the function symbols in Σ . This is trivially true for the case $i = 0$. Assume $\sim_{i-1}^{GA}$ and $\sim_{i-1}^{GB}$ are substitutable w.r.t. function symbols in Σ . For an arbitrary $f : s_1 \dots s_n \rightarrow s \in \Sigma$ and $(a_j, a'_j) \in \sim_i^{GA}, j = 1, \dots, n$, there exist $(b_j, b'_j) \in \sim_{i-1}^{GB}$ such that $(a_j, b_j), (a'_j, b'_j) \in G$.

From the definition of correspondence,

$(f^A(a_1, \dots, a_n), f^B(b_1, \dots, b_n)), (f^A(a'_1, \dots, a'_n), f^B(b'_1, \dots, b'_n)) \in G$. Furthermore, $(f^B(b_1, \dots, b_n), f^B(b'_1, \dots, b'_n)) \in \sim_{i-1}^{GB}$ since $\sim_{i-1}^{GB}$ is substitutable w.r.t f by the induction hypothesis. Hence $(f^A(a_1, \dots, a_n), f^A(a'_1, \dots, a'_n)) \in \sim_i^{GA}$. This shows $\sim_i^A$ is substitutable w.r.t. the function symbols in Σ . By symmetry, $\sim_i^{GB}$ is substitutable too.

From the construction of $\sim^{GA}$ and $\sim^{GB}$ and the fact G is a correspondence, it is clear that $A / \sim^{GA} \cong B / \sim^{GB}$. $\square$

7.3 Inductionless induction

An equation is *inductively provable* in a specification if it can be proved from the axioms by using structure induction, which implies all ground instances of this equation are true in the initial model. When the set of axioms satisfies certain conditions, inductive properties can be proved by checking consistency.

Several conditions for an axiom set to admit the inductionless induction proof technique have been proposed in the past few years. They can be classified into two categories:

- The set of axioms contain the axiomatization of an equality predicate, called 'equality enrichment' of the original specification [Gog 80][MG 83].
- The set of function symbols can be partitioned into *free constructors* and *defined symbols* such that every ground term can be proved equal to a ground term built from constructors only [Mus 80][HH 82].

Either of these conditions implies completeness of the specification when a suitable base specification is chosen. It is easy to see that, for the second condition, we can let the base signature consist of those free constructors and base specification have no equations. Then the axiom set is sufficiently complete, hence is base-complete which is the same as complete since there is no new sort involved in this case.

A bit more work is needed for the first condition. The key construction of 'equality enrichment' method in [MG 83] is the *equality enriched* specification $SP^= = (\Sigma^=, E^=)$ of a given specification $SP = (\Sigma, E)$. $\Sigma^=$ is obtained from Σ by adding a new sort **newbool** with constants **newtrue** and **newfalse**, a binary operation $\equiv_s$: $s \ s \rightarrow \text{newbool}$ for each sort s of SP . $E^=$ is obtained from E by adding some equations such that, for each $t, t' \in T_\Sigma$, $E_\Sigma \vdash (t \equiv t') = \text{newtrue}$ iff $E \vdash t = t'$, $E_\Sigma \vdash (t \equiv t') = \text{newfalse}$ iff $E \not\vdash t = t'$, and $E \not\vdash \text{newtrue} = \text{newfalse}$. Note that $T_{E^=}^S = T_\Sigma^S$.

Let $\Sigma_{\text{newbool}} = (\{\text{newbool}\}, \{\text{newtrue}, \text{newfalse}\})$ and $SP_{\text{newbool}} = (\Sigma_{\text{newbool}}, \phi)$. Now, instead of taking $SP^=$ as an extension of SP , we can think of $SP^=$ as an extension of SP_{newbool} : $SP^= = (SP_{\text{newbool}}, (\Sigma^=, E^=))$. Then $SP^=$ is consistent by definition. A more interesting fact is that $SP^=$ is in fact relatively-complete with respect to SP_{newbool} .

Fact 7.3 *$SP^=$ is complete w.r.t. the base specification SP_{newbool} .*

Proof Note first that the only terms of sort **newbool** are **newtrue**, **newfalse** and $t \equiv t'$ where $t, t' \in T_{\Sigma}$. Since either $E \vdash t = t'$ or $E \not\vdash t = t'$, either $E^= \vdash (t \equiv t') = \text{newtrue}$ or $E^= \vdash (t \equiv t') = \text{newfalse}$, i.e. $SP^=$ is sufficiently-complete, hence also base-complete.

Now suppose $\{t = t'\} \cup E^=$ is consistent where $t, t' \in T_{\Sigma^=}$ and have sort other than **newbool**.

Since $T_{\Sigma^=}$ and T_{Σ} have the same elements in any sort other than **newbool**, $t, t' \in T_{\Sigma}$. We have $E^= \vdash (t \equiv t) = \text{newtrue}$ because $E \vdash t = t$. So

$$\begin{aligned} \{t = t'\} \cup E^= &\vdash (t \equiv t') = (t \equiv t) \\ &\vdash (t \equiv t') = \text{newtrue} \end{aligned}$$

If $E \not\vdash t = t'$, then $E^= \vdash (t \equiv t') = \text{newfalse}$. It follows $\{t = t'\} \cup E^= \vdash \text{newtrue} = \text{newfalse}$, a contradiction with the hypothesis. So it must be the case $E \vdash t = t'$, and $E^= \vdash t = t'$ follows. $\square$

8 conclusion

We have presented the notion of relative-completeness and showed that a specification is complete if and only if its initial and final models coincide; a base-complete specification can be uniquely extended to a complete one and the final model of the later is just the initial model of the former. We also introduced the notion of ‘quotient-isomorphism’ and proved that the complete lattice of all generated models of a base-complete specification can be isomorphically embedded into the complete lattice of all quotient-initial models of its maximal consistent extension. Based on these results, we propose to use quotient-initial models as the semantics for extended specifications, which subsumes three different approaches to the specification of abstract data types, namely the initial, final, and generated model approaches, very nicely.

We confined the theoretical development in this paper to the institution of equations and total models. It should not be difficult to generalize our results to the institution of conditional equations and partial models.

Acknowledgements

The author is grateful to Paul Tailor and Andrzej Tarlecki for their careful reading of the draft version of this paper and many helpful comments and suggestions.

References

- [ADJ 78] Goguen, J.A., Thatcher, J.W. and Wagner, E.G., An initial algebra approach to the specification, correctness, and implementation of abstract data types. *Current Trends in Programming Methodology. Vol. 4: Data Structuring* (R.T. Yeh, ed.), Prentice-hall, pp. 80-149 (1978).
- [BG 77] Burstall, R.M. and Goguen, J.A., Putting theories together to make specifications. *Proc. 5th IJCAI*, Cambridge, Massachusetts, pp. 1045-1058 (1977)
- [BG 80] Burstall, R.M. and Goguen, J.A., The semantics of Clear, a specification language. *Proc. of Advanced Course on Abstract Software Specifications*, LNCS 86, pp.292-332
- [BW 84] Broy, M. and Wirsing, M., A systematic study of models of abstract data types. *TCS 33*, pp. 139-174 (1984)
- [CIP 85] Bauer, F.L. *et al*, *The Wide Spectrum Language CIP-L*. LNCS 183. (1985)
- [Ehr 82] Ehrich, H.-D., On the theory of specification, implementation, and parametrization of abstract data types. *JACM 29*, 1, pp.206-227 (1982)
- [EK 82] Ehrig, H. and Kreowski, H.-J., Parameter passing commutes with implementation of parameterized data types. *Proc. 9th Intl. Colloq. on Automata, Languages and Programming*, LNCS 140, pp.197-211
- [EKMP 82] Ehrig, H., Kreowski, H.-J., Mahr, B. and Padawitz, P., Algebraic implementation of abstract data types. *TCS 20*, pp.209-263 (1982)
- [EM 85] Ehrig, H., Mahr, B., *Fundamentals of Algebraic Specification I: Equations and Initial Semantics*. EATCS Monograph on Theoretical Computer Science, Springer.
- [EWT 83] Ehrig, H., Wagner, E.G. and Thatcher, J.W., Algebraic specifications with generating constraints. *Proc. 10th Intl. Colloq. on Automata, Languages and Programming*, LNCS 154, pp.188-202
- [Gan 83] Ganzinger, H., Parameterized specifications: parameter passing and implementation. *TOPLAS 3*, pp.318-364 (1983)
- [GDLE 84] Gogolla, M., Drosten, K., Lipect, U. and Ehrich, H.-D., Algebraic and operational semantics of specifications allowing exceptions and errors. *TCS 34*, pp.289-313 (1984)
- [GGM 76] Girratana, V., Gimona, F. and Montanari, U., Observability concepts in abstract data type specification. *Proc. 5th Intl. Symp. on Mathematical Foundations of Computer Science*, LNCS 45, pp.576-587.
- [GH 78] Guttag, J.V. and Horning, J.J., The algebraic specification of abstract data types. *Acta Informatica 10*, pp.25-52 (1978)
- [GB 84] Goguen, J.A. and Burstall, R.M., Introducing institutions. *Proc. Logics of Programming Workshop* (E. Clarke and D. Kozen, eds.), Carnegie-Mellon University. LNCS 164, pp.221-256

- [GJM 85] Goguen, J.A., Jouannaud, J.-P. and Meseguer, J., Operational semantics for order-sorted algebra. *Proc. 12th Intl. Colloq. on Automata, Languages and Programming*, LNCS 194, pp.221-231
- [GM 81] Goguen, J.A. and Meseguer, J., Completeness of many-sorted equational logic. *SIGPLAN Notices* 16 (1981)
- [GM 82] Goguen, J.A. and Meseguer, J., Universal realization, persistent interconnection and implementation of abstract modules. *Proc. 9th Intl. Colloq. on Automata, Languages and Programming*, LNCS 140, pp.265-281. (1982)
- [Gog 77] Goguen, J.A., Abstract errors for abstract data types. *Proc. IFIP Working Conf. on the Formal Description Programming Concepts*, New Brunswick, New Jersey. (1977)
- [Gog 80] Goguen, J.A., How to prove inductive hypothesis without induction. *Proc. 5th conf. on Automated Deduction*, LNCS 87, pp.356-372 (1980)
- [Gra 79] Gratzner, H., Universal algebra. Von Nostran, Princeton.
- [Hoa 72] Hoare, C.A.R., Proving correctness of data representation. *Acta Informatica* 1, pp.271-281 (1972)
- [HH 82] Huet, G. and Hullot, J.M., Proofs by induction in equational theories with constructors. *JCSS* 25 pp.239-266 (1982)
- [Kam 83] Kamin, S., Final data types and their specification. *TOPLAS* 5, pp.97-121 (1983)
- [LM 84] Lassez, J.-L. and Maher, M.J., Closures and fairness in the semantics of programming logic. *TCS* 29, No. 1-2, (1984)
- [Llo 84] Lloyd, J.W., Foundations of logic programming. Springer-Verlag. (1984)
- [Lin 86] Lin Huimin, A uniform framework for the specification of abstract data types. Ph.D. thesis (in Chinese), Institute of Software, Academia Sinica. (1986)
- [Mil 77] Milner, R., Fully abstract models of typed λ -calculi. *TCS* 4, pp.1-22 (1977)
- [MG 83] Meseguer, J. and Goguen, J.A., Initiality, induction and computability. *Algebraic Methods in Semantics*, (M. Nivat and J. Reynolds, eds.), Cambridge Univ. Press, pp.459-541
- [Mus 80] Musser, D.R., On proving inductive properties of abstract data types. *Proc. 7th POPL* (1980)
- [Rei 81] Reichel, H., Behavioural equivalence — a unifying concept for initial and final specification methods. *Proc. 3rd Hungarian Computer Science Conference*, Budapest, pp.27-39
- [ST 86a] Sannella, D.T. and Tarlecki, A., Specifications in an arbitrary institution. Report CSR-184-85, Dept. of Computer Science, Univ. of Edinburgh.

- [ST 86b] Sannella, D.T. and Tarlecki, A., On observational equivalence and algebraic specification. Report CSR-172-84, Dept. of Computer Science, Univ. of Edinburgh.
- [SW 82] Sannella, D.T. and Wirsing, M., Implementation of parameterized specifications. *Proc. 9th Intl. Colloq. on Automata, Languages and Programming*, LNCS 140, pp.473-488
- [Sch 83] Schoett, O., A theory of program modules, their specification and implementation (extended abstract). Report CSR-155-83, Dept. of Computer Science, Univ. of Edinburgh.
- [Sch 87] Schoett, O., Data abstraction and the correctness of modular programming. Ph.D. thesis, Dept. of Computer Science, Univ. of Edinburgh.
- [Tar 85] Tarlecki, A., On the existence of free models in abstract algebraic institutions. *TCS* 37, pp.269-304 (1985)
- [WB 80] Wirsing, M. and Broy, M., Abstract data types as lattices of finitely generated models. *9th MFCS*, LNCS 88, pp.673-685 (1980)
- [CIP 83] Wirsing, M., Pepper, P., Partsch, H., Dosch, W. and Broy, M., On hierarchies of abstract data types. *Acta Informatica* 20, pp.1-33 (1983)
- [Wan 79] Wand, M., Final algebra semantics and data type extensions. *JCSS* 19, pp.27-44 (1978)
- [Wan 82] Wand, M., Specification, models, and implementations of data abstraction. *TCS* 20, pp.3-32 (1982)

**Copyright © 1987, Laboratory for Foundations of Computer Science,
University of Edinburgh. All rights reserved.**

**Reproduction of all or part of this work
is permitted for educational or research use
on condition that this copyright notice is
included in any copy.**