stichting
mathematisch
centrum

MC

AFDELING INFORMATICA
(DEPARTMENT OF COMPUTER SCIENCE)

IW 183/81 NOVEMBER

J.A. BERGSTRA & J.W. KLOP

ALGEBRAIC SPECIFICATIONS FOR PARAMETRIZED DATA TYPES WITH MINIMAL PARAMETER AND TARGET ALGEBRAS

Preprint

kruislaan 413 1098 SJ amsterdam

Printed at the Mathematical Centre, 413 Kruislaan, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

1980 Mathematics subject classification: 03D45, 03D80, 68B15

ACM-Computing Reviews-category: 4.34

Algebraic specifications for parametrized data types with minimal parameter and target algebras *)

by

J.A. Bergstra**)& J.W. Klop

ABSTRACT

We conceive a parametrized data type as a partial functor ϕ: ALG (Σ) $\rightarrow$ ALG (Δ), where Δ is a signature extending Σ and ALG (Σ) is the class of minimal Σ-algebras which serve as parameters.

We focus attention on one particular method of algebraically specifying parametrized data types: finite specifications with conditional equations using auxiliary sorts and functions provided with initial algebra semantics.

We introduce the concept of an effective parametrized data type. A satisfactory adequacy result is then obtained: each effective parametrized data type possesses a finite algebraic specification under initial semantics.

KEYWORDS & PHRASES : *initial algebra specification, parametrized data type, semi- computable data type*

---

*) This report will be submitted for publication elsewhere.

**) Department of Computer Science, University of Leiden, Wassenaarseweg 80 2300 RA Leiden, The Netherlands

INTRODUCTION

The mathematical theory of parametrized data types was initially investigated in ADJ [13], [6], LEHMANN & SMYTH [10], KAPHENGST & REICHEL [9] and EHRICH [5]. Central topics in these studies are specification methods and the correctness problem for specifications and parameter passing mechanisms.

Reading through the growing litterature on parametrized data types one observes small but important differences between the basic definitions used by various authors; these variations resulting from differences in aims as well as from differences concerning the general points of view.

Obviously this situation entails a difficulty for the theoretical development of the subject. Rather than aiming at a unified theoretical framework it is our intention to consider one single specification method and to investigate that one in depth. This method is: initial algebra specifications with conditional equations using auxiliary sorts and functions.

The relevance of our results should not only be measured against the importance of the specification method that we analyze; it also indicates a style of investigating specification mechanisms for data types in general. The main idea is to connect specification methods to recursion theoretic concepts; similar results for abstract data type specification were obtained in BERGSTRA & TUCKER [2] and [3] .

A parametrized data type will be a partial functor ϕ: $ALG(\Sigma) \to ALG(\Delta)$, for some signatures Σ,Δ with $\Sigma \subseteq \Delta$. Here $ALG(\Gamma)$ denotes the class of all *minimal* algebras of signature Γ. (Remark on terminology: BURSTALL & GOGUEN [4] call $A \in ALG(\Gamma)$ an algebra 'without junk'.)

Further, ϕ is called *persistent* if $\phi(A)$ is an expansion of A for all $A \in Dom(\phi)$. Apart from the requirement that parameter algebras be minimal these definitions correspond to the original ones in ADJ [13].

All the constructions and arguments in the sequel will be *modulo isomorphism* of the minimal algebras we are dealing with. (Alternatively, one may consider $ALG(\Sigma)$, the class of minimal Σ-algebras, as consisting of *term* algebras, i.e. quotients of the free term algebra over Σ.) In this way we get around the difference between 'persistent' and 'strongly persistent' from ADJ [13]. For generalizations of our results however, a more sophisticated approach of this issue will be required.

Keeping in mind that the application of a parametrized data type on a parameter algebra is to be effectively performed in a computational process, the following class of *effective* parametrized data types seems to be of intrinsic importance. A parametrized data type ϕ is called effective iff there exists a computable transformation (γ,ε) that transforms a finite input specification (Σ',E') for a parameter algebra A into a finite specification $(\gamma(\Sigma',E'), \varepsilon(\Sigma',E')) = (\Sigma'',E'')$ for a target algebra $\phi(A)$. In both cases the specifications are allowed to use auxiliary sorts and functions.

An attractive transformation mechanism for specifications is the following one:

$$(\gamma(\Sigma',E'), \varepsilon(\Sigma',E')) = (\Sigma' \cup \Gamma, E' \cup E)$$

for some fixed finite specification (Γ,E). If such (Γ,E) can be found, the parametrized data type ϕ is said to have a *finite algebraic specification.*

Our main interest is the following question: to what extent are algebraic specifications available for effective parametrized data types. For this question we are interested in parametrized data types with a domain consisting of *semi-computable* algebras only, because other algebras have no finite specification. We are then able to prove the following adequacy theorem (where $SCA(\Sigma)$ denotes the class of semi-computable Σ-algebras):

THEOREM 3.1. *Let* ϕ: $ALG(\Sigma) \to ALG(\Delta)$ *be a persistent parametrized data type such that* $Dom(\phi) = ALG(\Sigma,E) \cap SCA(\Sigma)$ *for some finite* E. *Then* ϕ *is effective iff it has a finite algebraic specification.*

The proof is quite involved and uses a detour via an auxiliary notion, viz. that of a *(effectively) continuous* parametrized data type. A continuous parametrized data type ϕ can be represented by an element F in the Graph model $P\omega$ for the λ-calculus; an effectively continuous one by a recursively enumerable $F \in P\omega$. Now it turns out that a parametrized data type has a (finite) algebraic specification iff it is (effectively) continuous.

For further information about parametrized data types the reader is referred to [7], [8] and [14].

1. SPECIFICATION OF PARAMETER AND TARGET ALGEBRAS

In this section we will collect several definitions of preliminary notions and some facts about them.

1.1. Algebras.

A *signature* Σ is a triple consisting of three listings, one of sorts, one of functions and one of constants.

EXAMPLE. Σ: sorts INT , BOOL
functions SUC: INT $\to$ INT , $\neg$: BOOL $\to$ BOOL
constants $0 \in$ INT , true $\in$ BOOL.

Thus Σ determines the type of constants and functions declared in it. The meaning of $\Sigma \subseteq \Gamma$, $\Sigma \cup \Gamma$, $\Sigma \cap \Gamma$ is clear.

A Σ-*algebra* A consists of a non-empty set A_s for each sort s in Σ and functions $f^A : A_{s_1} \times \ldots \times A_{s_k} \to A_s$ for each function name $f \in \Sigma$ of type $s_1 \times \ldots \times s_k \to s$ and a constant $c^A \in A_s$ for each constant name c of type s in Σ.

For each sort $s \in \Sigma$ there are variables x^s_i , $i \in \omega$. The sets $Ter_s(\Sigma)$ of *terms* for sort $s \in \Sigma$ are defined by the following simultaneous induction. For each s:

(i) the constants of sort s are in $Ter_s(\Sigma)$;

(ii) $x^s_i \in Ter_s(\Sigma)$, $i \in \omega$;

(iii) if $\tau_j \in Ter_{s_j}(\Sigma)$, $j = 1,\ldots,k$, and $f \in \Sigma$ is a function of type $s_1 \times \ldots \times s_k \to s$ then $f(\tau_1,\ldots,\tau_k) \in Ter_s(\Sigma)$

Furthermore, $Ter(\Sigma) = \bigcup \{Ter_s(\Sigma) \mid s \text{ in } \Sigma\}$.

A *closed* term contains no variables. $Ter^c(\Sigma)$ is the set of closed Σ-terms. An *equation* (of sort s) is an expression of the form $\tau = \tau'$ where $\tau,\tau' \in Ter_s(\Sigma)$. A closed equation is an equation between closed terms. A *conditional equation* is a construct of the form

$$\tau_1 = \tau_1' \wedge \ldots \wedge \tau_k = \tau_k' \to \tau = \tau'$$

where $\tau_i , \tau_i' \in Ter_{s_i}(\Sigma)$, $i = 1, \ldots, k$ and $\tau,\tau' \in Ter_s(\Sigma)$ for some s_i,s.

The *free term algebra* $T(\Sigma)$ is obtained by taking as A_s (see above) the sets $Ter^c_s(\Sigma)$ and interpreting functions and constants 'by themselves'.

A Σ-algebra A is *minimal* if it has no proper Σ-subalgebras. If $\Gamma \supseteq \Sigma$ and A is some Γ-algebra, then $A|_\Sigma$ is the *reduct* of A of signature Σ which results by forgetting sorts, constants and functions not named in Σ. By $\langle A \rangle_\Sigma$ we denote the minimal Σ-subalgebra of $A|_\Sigma$. If $A|_\Sigma = \langle A \rangle_\Sigma = B$, we write $(A)_\Sigma = B$ and call A an *enrichment* of B.

With $ALG(\Sigma)$ we denote the class of minimal Σ-algebras. For a set E of conditional equations, $ALG(\Sigma,E)$ denotes the class of algebras $A \in ALG(\Sigma)$ with $A \models E$.

To each $A \in ALG(\Sigma)$ we can associate the *congruence* $\equiv_A$, that is the set of all closed equations true in A. Note that $A \cong T(\Sigma)/\equiv_A$ (A is isomorphic to the factor algebra obtained from the free term algebra by dividing out its congruence).

If $K \subseteq ALG(\Sigma)$, then $I(K)$ denotes the initial algebra of K, if it exists. (This is the algebra A from which all $B \in K$ are homomorphic images; A is determined up to isomorphism.)

1.2. Recursion theory and coding.

We use the notation W_z (of ROGERS [11]) for recursively enumerable (r.e) subsets of ω; $z \in \omega$ is called an r.e. -index.

Often we will use a bijective and effective *coding* $\ulcorner\ \urcorner : S \to \omega$ for a set S of syntactic constructs, e.g. $S = Ter^c(\Sigma)$. *Decoding* $\llcorner\ \lrcorner : \omega \to S$ is given by the inverse function. It is left to the reader to give a detailed construction of $\ulcorner\ \urcorner$. If $T \subseteq S$, then $\ulcorner T \urcorner = \{\ulcorner t \urcorner \mid t \in T\}$; likewise $\llcorner A \lrcorner$, for $A \subseteq \omega$, is defined.

Let $A \in ALG(\Sigma)$. Then A is called *semi-computable* iff $\ulcorner \equiv_A \urcorner$ is r.e. (iff $\exists z\, \ulcorner \equiv_A \urcorner = W_z$). The set of semi-computable minimal Σ-algebras is denoted by $SCA(\Sigma)$.

Let $\ulcorner\ \urcorner : TER^c(\Sigma) \times Ter^c(\Sigma) \to \omega$ be a bijective coding of all closed Σ-equations, with $\llcorner\ \lrcorner$ as decoding function. Now an arbitrary $\llcorner W_z \lrcorner$ need not yet be a congruence; it is after closure under logical derivability:

$\overline{\llcorner W_z \lrcorner}$.

Coding again it is not hard to see that $\overline{\lfloor W_z \rfloor} = W_{c(z)}$ for some recursive $c : \omega \to \omega$. So $W_{c(z)}$ codes a congruence, for all $z \in \omega$. (See also the diagram in section 1.3.)

1.3. Initial algebra specifications.

Let $A \in ALG(\Sigma)$, and $\Sigma' \supseteq \Sigma$. Then (Σ',E') is a *specification of* A *using auxiliary sorts and functions* if $A = (I(ALG(\Sigma',E')))_\Sigma$. For brevity we will use the notation: $(\Sigma',E')_\Sigma = A$. To employ in diagrams, we use the alternative notation:

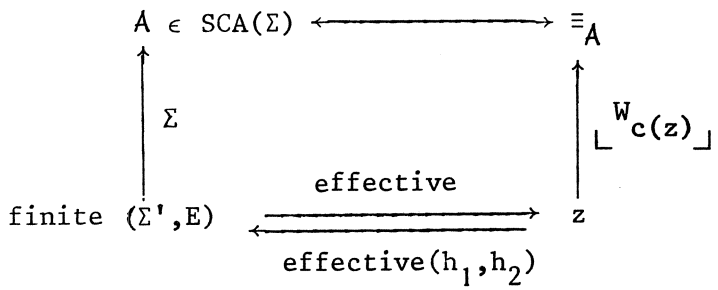
$$(\Sigma',E') \xrightarrow{\Sigma} A \ .$$

Note that $I(ALG(\Sigma',E'))$ always exists. However, $(I(ALG(\Sigma',E')))_\Sigma$ is not for all (Σ',E') and $\Sigma' \supseteq \Sigma$ defined (see the definition of enrichment in 1.1). Note that if E' is finite, $I(ALG(\Sigma',E')) \in SCA(\Sigma')$. In fact we have:

1.3.1. LEMMA. $A \in SCA(\Sigma) \iff A = (\Sigma',E')_\Sigma$ *for some* $\Sigma' \supseteq \Sigma$ *and finite* E'.

This is proved in BERGSTRA & TUCKER [1] . In fact it is proved there that from an r.e.-index z for $\overline{\equiv_A}$ one can *uniformly* find a finite (Σ',E') specifying A ; see the diagram below.

Finite specifications (Σ',E') for A can be thought of as 'indices' just like z is an r.e.-index for $\equiv_A$ $(= W_{\lfloor z \rfloor})$ after coding. Indeed, the following diagram asserts that both kinds of indices can effectively be translated into each other:



2. PARAMETRIZED DATA TYPES, DESCRIPTIONS AND SPECIFICATIONS

In this section we explain our definition of a parametrized data type, and explain what it means for a parametrized data type to be: *effectively given, algebraically specified, continuous* or *effectively continuous*.

2.1 Parametrized data types.

A parametrized data type is a partial functor ϕ: ALG(Σ) $\to$ ALG(Δ) where $\Sigma \subseteq \Delta$, i.e.

$$\begin{array}{ccc} A & \longrightarrow & \phi(A) \\ \text{hom.}\downarrow \alpha & & \exists\ \text{hom.}\downarrow \beta \\ B & \longrightarrow & \phi(B) \end{array}$$

which satisfies the following condition : for each $A \in \mathrm{Dom}(\phi)$ there is a surjective homomorphism α: $A \to \phi(A)|_\Sigma$.

If, moreover, for each $A \in \mathrm{Dom}(\phi)$ we have: $A \cong \phi(A)|_\Sigma$ then ϕ is *persistent*.

2.2. ϕ is effective given (ϕ is effective)

if $\mathrm{Dom}(\phi) \subseteq \mathrm{SCA}(\Sigma)$ and there is a pair (γ,ε) of computable operations, acting on finite specifications, that produces a specification $(\gamma(\Sigma',E'),\ \varepsilon(\Sigma',E'))$ of $\phi(A)$ for each specification (Σ',E') of some $A \in \mathrm{Dom}(\phi)$.

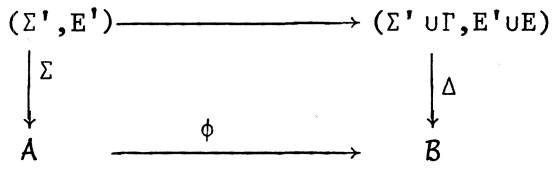
In a diagram :

$$\begin{array}{lccc} & \text{finite } (\Sigma',E') & \xrightarrow{\text{comp.}(\gamma,\varepsilon)} & (\gamma(\Sigma',E'),\ \varepsilon(\Sigma',E')) = (\Sigma'',E''),\ \text{finite} \\ & \Big\downarrow \Sigma & & \Big\downarrow \Delta \\ \text{semi-computable} & A \in \mathrm{Dom}(\phi) & \xrightarrow{\qquad\qquad \phi \qquad\qquad} & B,\ \text{semi-comp.} \end{array}$$

In a different notation: $\phi((\Sigma',E')_\Sigma) = (\gamma(\Sigma',E'),\ \varepsilon(\Sigma',E'))_\Delta$.

2.3. ϕ has an algebraic specification

if there is a specification (Γ,E) such that for all $A \in Dom(\phi)$:



If (Γ,E) is finite, then ϕ has a finite algebraic specification; in that case $\phi \upharpoonright SCA(\Sigma)$ is effectively given with $\gamma(\Sigma',E') = \Sigma'\cup\Gamma$ and $\varepsilon(\Sigma',E') = E'\cup E$. Here it is required that $\Sigma'\cap\Gamma \subseteq \Sigma$.

Notation : $\phi \subseteq (\Gamma,E)^{\Sigma}_{\Delta}$; so the diagram states:

$$(\Gamma,E)^{\Sigma}_{\Delta}\ (\Sigma',E')_{\Sigma} = (\Sigma'\cup\Gamma,E'\cup E)_{\Delta}\ .$$

Note the following composition rule (provided $\Gamma'\cap\Gamma = \Delta$) :

$$(\Gamma',F)^{\Delta}_{\Pi} \circ (\Gamma,E)^{\Sigma}_{\Delta} = (\Gamma'\cup\Gamma,F\cup E)^{\Sigma}_{\Pi}\ .$$

2.4. Representing parametrized data types in reflexive domains.

2.4.1. Let $\ulcorner\ \urcorner_\Gamma$ be a bijective coding of closed Γ-equations, and $\llcorner\ \lrcorner_\Gamma$ the corresponding decoding. We will omit the Γ when no confusion is likely to arise.

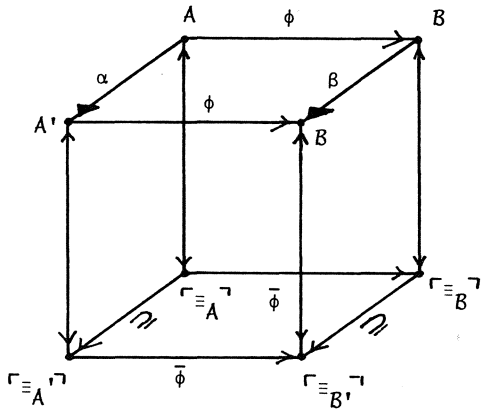
For a parametrized data type ϕ: $ALG(\Sigma) \to ALG(\Delta)$, let

$$\ulcorner Dom(\phi)\urcorner = \{\ \ulcorner \equiv_A \urcorner^{\Sigma}\ |\ A \in Dom(\phi)\}\quad \text{and}$$

$$\ulcorner Range(\phi)\urcorner = \{\ulcorner \equiv_B \urcorner^{\Delta}\ |\ B \in Range\ (\phi)\}.$$

The mapping $\bar{\phi} : \ulcorner Dom(\phi)\urcorner \to \ulcorner Range(\phi)\urcorner$ is introduced by

$$\bar{\phi}(\ulcorner \equiv_A \urcorner) = \ulcorner \equiv_{\phi(A)} \urcorner .\ (\text{See diagram.})$$



2.4.2. A reflexive domain. The Graph model Pω is the structure consisting of the powerset of ω and an application operator · on it. Application is defined as follows: for A,B ∈ Pω,

A·B = {m|∃ n ∈ ω (n,m) ∈ A & $D_n \subseteq B$} where (,): ω × ω → ω is a bijective and effective pairing function and D_n is the finite set with 'canonical index' n defined as follows: $D_0 = \emptyset$; if $n = 2^{a_1} + \ldots + 2^{a_k}$, $a_1 < \ldots < a_k$, then $D_n = \{a_1, \ldots, a_k\}$.

A mapping F: Pω → Pω is *continuous* if for all X ∈ Pω: $F(X) = \bigcup\{F(D_n) \mid D_n \subseteq X\}$. For the next Lemma, see SCOTT [12].

2.4.2.1. LEMMA. *Let* F: Pω → Pω. *Then*:
F *is continuous* ⟺ ∃F ∈ Pω ∀X ∈ Pω F(X) = F·X.

2.4.2.2. DEFINITION. (i) The parametrized data type φ is *continuous* if $\overline{\varphi}$ is the restriction to ⌜Dom(φ)⌝ of some continuous mapping F: Pω → Pω.
(ii) Moreover, φ is called *effectively continuous* if $\overline{\varphi}$ is the restriction of a continuous F which is represented in Pω by an r.e. element F ∈ Pω. (I.e. F is an *enumeration operator*, in the sense of ROGERS [11].)

2.4.2.3. Write RE for the set of r.e. subsets of Pω. Let Φ: RE → RE . Then Φ is called *effective* if for some computable f:

$$\forall z \quad \Phi(W_z) = W_{f(z)} \ .$$

We need the following version of the Theorem of Myhill and Shepherdson (see ROGERS [11]), as stated in SCOTT [12]:

2.4.2.4. THEOREM. *If* Φ: RE $\to$ RE *is effective, then for some r.e. element* F of Pω:

$$\forall X \in RE \quad \Phi(X) = F \cdot X \ .$$

Consequently Φ as in the Theorem can be extended to a continuous operator (viz. $\lambda X.\ F \cdot X$). On the other hand of course: if $F \in RE$, then $\lambda X \in RE.\ F \cdot X$ is effective.

3. SPECIFICATION THEOREMS

The main result of this paper is Theorem 3.1 which essentially asserts that effective parametrized data types have finite specifications, provided their domain is reasonably well-behaved. We expect that 3.1 (ii)$\Longleftrightarrow$(iii) will have many generalizations; for instance, removing the condition that input algebras are minimal seems quite worth-wile. Other specification methods, such as working with requirements (see EHRIG [7])or with final algebras, lead to similar questions.

Theorems 3.2 and 3.3 provide exact characterizations of the persistent parametrized data types that can be specified, without any condition on the domains involved.

3.1. THEOREM. *Let* ϕ: ALG(Σ) $\to$ ALG(Δ) *be a persistent parametrized data type with* Dom (ϕ) = ALG(Σ,E) $\cap$ SCA(Σ), *for some finite E. Then the following are equivalent:*

(i) ϕ *is effectively continuous;*

(ii) ϕ *possesses a finite algebraic specification;*

(iii) ϕ *is effective.*

3.2. THEOREM. *Let* ϕ: ALG(Σ) $\to$ ALG(Δ) *be a persistent parametrized data type. Then the following are equivalent:*

(i) ϕ *is continuous;*

(ii) ϕ *has an algebraic specification.*

3.3. THEOREM. *Let* ϕ: ALG(Σ) $\to$ ALG(Δ) *be a persistent parametrized data type. Then the following are equivalent:*

(i) ϕ *is effectively continuous;*

(ii) ϕ *has a finite algebraic specification.*

Since the proofs are rather involved we will make some remarks about their structure. (See also fig. 1 below.) First we will prove the continuity properties for all three theorems; i.e. all upward arrows in fig. 1,2,3. This is done in section 4. In section 5 we prove an important trio of lemma's enabling us to prove (i) $\Rightarrow$ (ii) for the three theorems above. The proofs of these specification lemma's require some theory of '*lifting of specifications*' which is of a technical nature. In order not to obscure the main line of the arguments, this technical part is given in an Appendix. Section 6 contains the combination of the three specification lemma's which yields the remaining parts of the proofs of Theorems 3.1, 3.2 and 3.3.

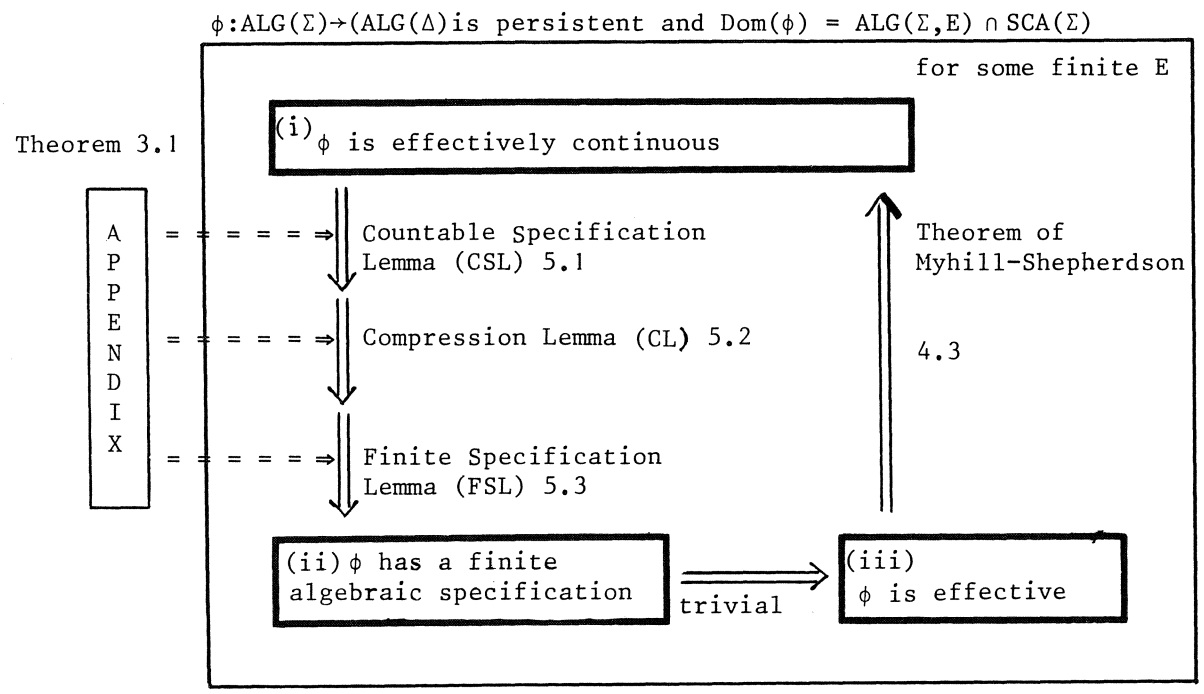


fig. 1

Theorem 3.2

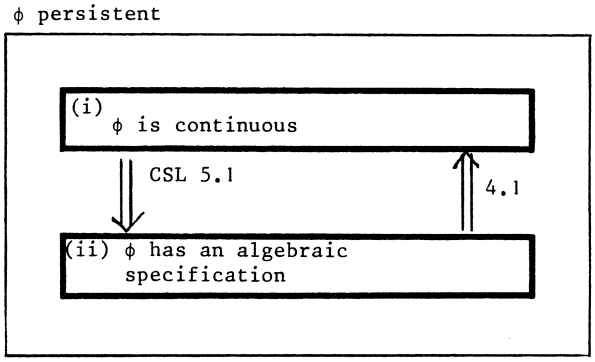


fig.2

Theorem 3.3

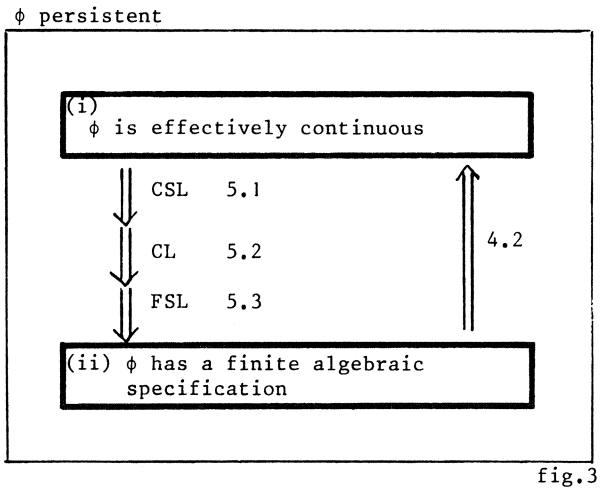


fig.3

4. PROVING CONTINUITY

We will now prove (iii) ⇒ (ii) of Theorem 3.1 and (ii) ⇒ (i) of Theorems 3.2, 3.3. First the easier two implications:

4.1. Proof of Theorem 3.2 (ii) ⇒ (i).

Let $\ulcorner\ \urcorner$ and $\llcorner\ \lrcorner$ be bijective coding and decoding functions for closed Σ - equations, and likewise $\llbracket\ \rrbracket$, $\llparenthesis\ \rrparenthesis$ for closed Δ-equations.

Suppose that ϕ has a specification, say (Γ,F). So $\phi(A) = (\Gamma,F)^{\Sigma}_{\Delta}(A)$, for $A \in \mathrm{Dom}(\phi)$. Noting that $A = (\Sigma,\equiv_A)_\Sigma$, we have

$$\phi(A) = (\Gamma,F)^{\Sigma}_{\Delta}(\Sigma,\equiv_A)_\Sigma = (\Gamma \cup \Sigma, F \cup \equiv_A)_\Delta .$$

Now let $A = \{(n,m) \mid F \cup \lfloor D_n \rfloor \vdash \llbracket m \rrbracket\}$, $A \in P\omega$. Then for $A \in \mathrm{Dom}(\phi)$:

$$A \cdot \ulcorner \equiv_A \urcorner = \{m \mid \exists D_n \subseteq \ulcorner \equiv_A \urcorner\ (n,m) \in A\} =$$

$$\{m \mid \exists D_n \subseteq \ulcorner \equiv_A \urcorner\ F \cup \lfloor D_n \rfloor \vdash \llbracket m \rrbracket\} = \{m \mid F \cup \equiv_A \vdash \llbracket m \rrbracket\} =$$

$$\{\llbracket e \rrbracket \mid F \cup \equiv_A \vdash e\} = \{\llbracket e \rrbracket \mid (\Gamma \cup \Sigma, F \cup \equiv_A)_\Delta \models e\} = \llbracket \equiv_{\phi(A)} \rrbracket =$$

$$\overline{\phi}(\ulcorner \equiv_A \urcorner)\ .$$

Hence ϕ is continuous (by Def. 2.4.2.2 and Lemma 2.4.2.1). □

4.2. Proof of Theorem 3.3 (ii) ⇒ (i).

If in the above proof F is finite, then obviously A is r.e. . Hence ϕ is effectively continuous. □

4.3. Proof of Theorem 3.1 (iii) ⇒ (ii).

Let (γ,ε) be an effective transformation of specifications that describes ϕ. Consider $\overline{\phi}$. We will construct an effective operator (see 2.4.2.3) δ: RE → RE that extends $\overline{\phi}$. Then it follows by the Theorem of Myhill & Shepherdson (2.4.2.4) that δ can be extended to an enumeration operator (2.4.2.2(ii)), which immediately implies that ϕ is effectively continuous.

In order to define δ, consider the domain $\mathrm{ALG}(\Sigma,E) \cap \mathrm{SCA}(\Sigma)$ of ϕ . Let $W_{d(z)}$ be the coded congruence of an algebra in $\mathrm{ALG}(\Sigma,E) \cap \mathrm{SCA}(\Sigma)$ which is generated by W_z (cfr. $W_{c(z)}$ in diagram in 1.3; there $E = \emptyset$). To be precise, let d be a recursive function such that for all z:

$$\lfloor W_{d(z)} \rfloor = \{e \mid e \text{ is a closed } \Sigma\text{-equation } \&\ E \cup \lfloor W_z \rfloor \vdash e\}\ .$$

Such a function d exists because E is finite.

Further, let (h_1,h_2) be as in the diagram in 1.3, and let $(\Sigma'(z)\ ,\ E'(z)) = (h_1(d(z))\ ,\ h_2(d(z))$. Now define:

$$\delta(W_z) = \{\ulcorner e \urcorner \mid (\gamma(\Sigma'(z),E'(z)),\ \varepsilon(\Sigma'(z),E'(z))) \vdash e,$$
$$e \text{ is a closed } \Delta \text{ - equation}\}$$

$$= W_{g(z)}$$

for an appropriate computable function g.

One easily verifies that δ is an effective operator. Moreover, δ extends $\bar{\phi}$: let $A \in \mathrm{Dom}(\phi)$ and $\ulcorner \equiv_A \urcorner = W_z$. Then $W_z = W_{d(z)}$ and thus $(\Sigma'(z),E'(z) \xrightarrow{\Sigma} A$ and $(\gamma(\Sigma'(z),E'(z)),\ \varepsilon(\Sigma'(z),E'(z))) \xrightarrow{\Delta} \phi(A)$ which implies $W_{g(z)} = \ulcorner \equiv_{\phi(A)} \urcorner$. Hence $\delta(W_z) = \bar{\phi}(\ulcorner \equiv_A \urcorner)$. □

5. THREE SPECIFICATION LEMMA'S

Since the proof of Theorem 3.1(ii) ⇒ (iii) is trivial and since Theorem 3.1 (i) ⇒ (ii) follows from the more general implication 3.3 (i) ⇒ (ii), it remains to establish (i) ⇒ (ii) for Theorems 3.2 and 3.3 . This is done as follows.

Given a continuous parametrized data type ϕ, we have an $F \in P\omega$ representing ϕ. Now the Countable Specification Lemma (5.1) transforms this F into a countable specification E_F for ϕ consisting of closed conditional equations. This proves already Theorem 3.2 (i) ⇒ (ii).

If moreover ϕ is effectively continuous, F is r.e. . Then the Finite Specification Lemma (5.3) is able to convert the countable specification E_F into a finite one; but first E_F has to be 'preprocessed' by the Compression Lemma (5.2) to an E_F' containing only closed conditional equations $e \to e'$ with precisely one condition.

5.1. COUNTABLE SPECIFICATION LEMMA. *Let* ϕ: ALG(Σ) → ALG(Δ) *be a persistent and continuous parametrized data type. Then* ϕ *has a specification* (Δ,E) *with* E *containing closed conditional equations only.*

If moreover ϕ *is effectively continuous, then* E *can be taken to be an r.e. set.*

PROOF. Let ϕ be continuous. Let $F \in P\omega$ represent $\bar{\phi}$(i.e. F extends $\bar{\phi}$) .

Let $\ulcorner\ \urcorner$, $\llcorner\ \lrcorner$, $\llbracket\ \rrbracket$ and $\mathbb{L}\ \mathbb{J}$ be as in 4.1 .

Now there is a nice correspondence between $(m,n) \in F$ and closed conditional equations, as follows: to each $(m,n) \in F$ we associate the conditional equation

$$e_{(m,n)} = M \llcorner D_m \lrcorner \rightarrow \mathbb{L} n \mathbb{J} \quad .$$

These closed conditional equations turn out to be the desired specification:

$$\phi \subseteq (\Delta, E_F)^{\Sigma}_{\Delta} \qquad (*)$$

where $E_F = \{ e_{(m,n)} \mid (m,n) \in F \}$.

We will now prove that (*) indeed holds. In order to do so, we need a proposition expressed in the following claim. There the following notation is used: if E is a set of conditional equations , E° is the set of all *closed equations* logically derivable from E.

CLAIM. *Let* ϕ,F *and* E_F *be as above. Then:*

(i) $A \in \mathrm{Dom}(\phi) \Rightarrow (E_F \cup \Xi_A)^{\circ} \subseteq \Xi_{\phi(A)}$,

(ii) *if* ϕ *is persistent:*

$A \in \mathrm{Dom}(\phi) \Rightarrow (E_F \cup \Xi_A)^{\circ} = \Xi_{\phi(A)}$.

Proof of the claim.

(i) is obvious from the construction of E_F .

(ii) It suffices to show that $\phi(A) \models E_F \cup \Xi_A$.
That $\phi(A) \models \Xi_A$ is obvious since $(\phi(A))_{\Sigma}$ is a homomorphic image of A . Also $\phi(A) \models E_F$; for, let $e_{(m,n)} \in E_F$. Assume $\phi(A) \models M \llcorner D_m \lrcorner$. Then also $(\phi(A))_{\Sigma} \models M \llcorner D_m \lrcorner$. By persistency $A = (\phi(A))_{\Sigma}$, hence $A \models M \llcorner D_m \lrcorner$. Now
$A \models M \llcorner D_m \lrcorner \iff$

$$
\begin{aligned}
&\lfloor D_m \rfloor \subseteq \equiv_A \quad \Leftrightarrow \\
&D_m \subseteq \ulcorner \equiv_A \urcorner \quad \Rightarrow \\
&n \in \llbracket \equiv_{\phi(A)} \rrbracket \quad \Leftrightarrow \\
&\llbracket n \rrbracket \in \equiv_{\phi(A)} \quad \Leftrightarrow \\
&\phi(A) \models \llbracket n \rrbracket \ .
\end{aligned}
$$

Therefore $\phi(A) \models M_{\lfloor D_m \rfloor} \to \llbracket n \rrbracket (= e_{(m,n)})$.

So if ϕ is persistent, then for $A \in \mathrm{Dom}(\phi)$:

$$(\Delta, E_F)^{\Sigma}_{\Delta} (\Sigma, \equiv_A)_{\Sigma} = (\Delta, E_F \cup \equiv_A)_{\Delta} = (\Delta, (E_F \cup \equiv_A)^{\circ})_{\Delta} = \text{(by the claim)}$$

$$(\Delta, \equiv_{\phi(A)})_{\Delta} = \phi(A).$$

Now (*) follows by the Standard Application Lemma (App. 7.2). □

In the next two lemma's the concept $(\Gamma', E') \trianglerighteq (\Gamma, E)$ (the specification (Γ', E') *is a lifting of* (Γ, E)) is employed. The precise definition and the proof of the 'Lifting Lemma' are given in the Appendix. The intuitive idea is simply that a lifting (Γ', E') of (Γ, E) is some kind of extension of the specification (Γ, E) such that they specify the same parametrized data types:

$$(\Gamma', E') \trianglerighteq (\Gamma, E) \Rightarrow (\Gamma', E')^{\Sigma}_{\Delta} = (\Gamma, E)^{\Sigma}_{\Delta} \ .$$

(In fact we must be slightly more precise - see the Appendix.)

5.2. COMPRESSION LEMMA.

Let (Γ, E) *be a specification with* E *containing closed conditional equations only. Then there is a lifting* (Γ', E') *of* (Γ, E) *with* E' *containing closed conditional equations of the form* $e \to e'$ *only.*

Moreover, if E *is r.e. then so is* E'.

PROOF. Consider the following extension $\Gamma \cup \Delta$ of Γ:
the signature Δ has sorts NAT , LINK
functions S: NAT → NAT
L: NAT × NAT → LINK
constants $0 \in$ NAT
We use the abbreviation $\underline{k}$ for the term $S^k(0)$ of sort NAT ($k \in \omega$).

Let $E = \{s_1 = t_1 \wedge \ldots \wedge s_{m_i} = t_{m_i} \to s_i' = t_i' \mid i \in \omega\}$ be a (not necessarily effective) enumeration of E, for some function $i \mapsto m_i$. We may suppose $m_i \geq 1$ (by prefixing a dummy condition if necessary).
Consider e_i: $s_1 = t_1 \wedge \ldots \wedge s_{m_i} = t_{m_i} \to s_i' = t_i'$ ($m_i \geq 1$). We will replace e_i by the set E_i of $m_i + 1$ conditional equations each having only one condition:

$$\begin{cases} s_1 = t_1 \to L(\underline{i},\underline{0}) = L(\underline{i},\underline{1}) \\ s_2 = t_2 \to L(\underline{i},1) = L(\underline{i},2) \\ \quad\vdots \qquad\qquad\qquad \vdots \\ s_{m_i} = t_{m_i} \to L(\underline{i},k-1) = L(\underline{i},\underline{k}) \\ L(\underline{i},\underline{0}) = L(\underline{i},\underline{k}) \to s_i' = t_i' \end{cases}$$

(Note that using these cond. equations:
$s_1 = t_1 \wedge \ldots \wedge s_{m_i} = t_{m_i} \to L(\underline{i},0) = L(\underline{i},1) = L(\underline{i},2) = \ldots = L(\underline{i},\underline{k}) \to s_i' = t_i'$.)
Now (Γ',E') will be $(\Gamma \cup \Delta, \bigcup_{i\in\omega} E_i)$. The verification that indeed $(\Gamma',E') \trianglerighteq (\Gamma,E)$ is left to the reader.

If E is r.e., it is not hard to see that E' is r.e. too .

5.3. FINITE SPECIFICATION LEMMA.

Let (Γ,E) *be a specification with* E *an r.e. set of conditional equations of the form* $e \to e'$.

Then (Γ,E) *has a lifting* (Γ',E') *with* E' *finite.*

PROOF. Let $E = \{e_i \to e_i' \mid i \in \omega\}$ be an effective enumeration of E. Let $E^{(s,t)}$ contain all the conditional equations in E of the form $\tau_1^s = \tau_2^s \to \tau_3^t = \tau_4^t$, for every pair of sorts (s,t) in Γ. So $E = \cup\{E^{(s,t)} \mid s,t \text{ sorts in } \Gamma\}$.

Let $\ulcorner\ \urcorner$, $\llcorner\ \lrcorner$ be bijective coding and decoding functions of the closed Γ - terms. Since E is r.e., each $E^{(s,t)}$ is r.e.; hence for each pair (s,t) there are recursive functions $g_i^{(s,t)}$, i=1, ...,4 such that

$$E^{(s,t)} = \{ \llcorner g_1(n) \lrcorner = \llcorner g_2(n) \lrcorner \to \llcorner g_3(n) \lrcorner = \llcorner g_4(n) \lrcorner \mid n \in \omega \}.$$

Now we could give the desired $(\Gamma',E') \trianglerighteq (\Gamma,E)$ at once; however for a better understanding, and as an anticipation of the proof of $\trianglerighteq$, we will consider first the following expansion of $A \in ALG(\Gamma)$.

First we define an algebra $\bar{E}$, determined by E. Let $^*: \Gamma \to \Gamma^*$ be a signature morphism making a disjoint copy of Γ. So to each constant c, resp. function f in Γ there corresponds c^* , f^* in Γ^*. We will extend * in the obvious way to $Ter(\Gamma)$.

The *signature* of $\bar{E}$, called $\Sigma_{\bar{E}}$, is:

$\Gamma^* \cup$ sorts : NAT

functions: $S: NAT \to NAT$

$G_i^{(s,t)}: NAT \to \Gamma^*$ (for each $s,t \in$ sorts(Γ) and $i = 1, \ldots, 4$)

constants : 0

(as before, $\underline{k}$ abbreviates $S^k(0)$, for $k \in \omega$.)

The *congruence* $\equiv_{\bar{E}}$ is generated by the recursive set of closed $\Sigma_{\bar{E}}$-equations:

$$G = \{G_i^{(s,t)}(\underline{k}) = \llcorner g_i^{(s,t)}(k) \lrcorner^* \mid k \in \omega,\ s,t \in \text{sorts } (\Gamma)\}$$

So $\bar{E} = (\Sigma_{\bar{E}},G)_{\Sigma_{\bar{E}}}$. Clearly $\bar{E}$ is a semi-computable (even a computable) algebra. Hence by Lemma 1.3.1 it has a *finite* specification (Δ,F) for some $\Delta \supseteq \Sigma_{\bar{E}}$ and F.

Next, $\bar{E}$ is "glued" to A , by means of homomorphisms $h^s: T(\Gamma^*) \to A$ (note that $(\bar{E})_{\Gamma^*} = T(\Gamma^*)$) for every sort s in Γ, satisfying the finite set H of equations:

$$H = \begin{cases} h^s(c^*) = c \text{ , for every } c \in \underline{\text{constants}}\ (\Gamma) \\ h^s(f^*(x_1, \ldots, x_k)) = f(h^{s_1}(x_1), \ldots, h^{s_k}(x_k)) \\ \text{for every } f \in \underline{\text{functions}}\ (\Gamma) \text{ of type } s_1 \times \ldots \times s_k \to s \end{cases}$$

(So the h^s remove the * of Γ^* - symbols.)
Let $A \oplus E$ be the result , see figure 4.

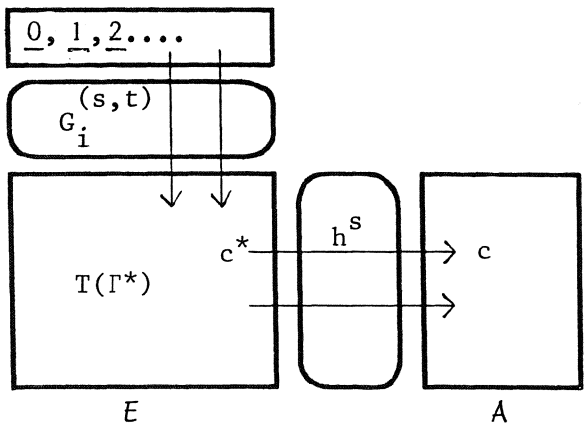


Fig. 4

Now consider the finite set of conditional equations
$\mathbb{E} = \{e^{(s,t)} \mid s,t \in \underline{\text{sorts}}\ (\Gamma)\}$, where

$$e^{(s,t)}\colon h^s(G_1^{(s,t)}(x)) = h^s(G_2^{(s,t)}(x)) \to h^t(G_3^{(s,t)}(x)) = h^t(G_4^{(s,t)}(x)).$$

Evidently, if $A \oplus E \models \mathbb{E}$, then $A \models E$. So using E the infinite E can be replaced by the finite $\mathbb{E}$.

It is now clear what the desired (Γ',E') , such that $(\Gamma',E') \trianglerighteq (\Gamma,E)$, should be:

$$(\Gamma',E') = (\Gamma \cup \Delta \cup \{h^s \mid s \in \underline{\text{sorts}}\ (\Gamma)\} \text{ , } \mathbb{E} \cup F \cup H).$$

The proof that $(\Gamma',E') \trianglerighteq (\Gamma,E)$ is routine; the expansion requirement (see Def. App. 7.3(iii)) is clearly fulfilled since every $A \in \mathrm{ALG}(\Sigma,E)$ can be expanded to $A \oplus E' \in \mathrm{ALG}(\Gamma',E')$ where $E' = I(\mathrm{ALG}(\Delta,F))$. □

6. PROOF OF THEOREM 3.2 (i) ⇒ (ii) AND 3.3 (i) ⇒ (ii).

Clearly 3.2 (i) ⇒ (ii) is a consequence of the Countable Specification Lemma (CSL) 5.1.

The other implication requires some argument. Let ϕ: ALG(Σ) → ALG(Δ) be persistent and effectively continuous. According to CSL 5.1 it has a specification (Δ,E) with E r.e. and containing closed conditional equations only. According to the Compression Lemma (5.2) this specification can be lifted to a specification (Γ,F) with F r.e. and containing closed conditional equations of the form e → e' only.

Then, using the Finite Specification Lemma (5.3), (Γ,F) is lifted to (Γ',F') with F' finite. By transitivity of lifting, (Γ',F') ≥ (Δ,E).

Finally, by the Lifting Lemma (App. 7.4) we may conclude from $\phi \subseteq (\Delta,E)^{\Sigma}_{\Delta}$ to $\phi \subseteq (\Gamma',F')^{\Sigma}_{\Delta}$, i.e. ϕ possesses a finite specification. □

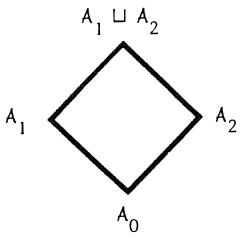
7. APPENDIX: LIFTINGS OF SPECIFICATIONS

Before we state the definition of lifting and prove its main property, we need some preparation.

7.1. JOINT EXPANSION LEMMA.

Let $A_i \in ALG(\Sigma_i)$, i = 0,1,2, *be such that* $\Sigma_1 \cap \Sigma_2 = \Sigma_0$ *and* $(A_1)_{\Sigma_0} = A_0 = (A_2)_{\Sigma_0}$.

Then there is a unique joint expansion $A_1 \sqcup A_2 \in ALG(\Sigma_1 \cup \Sigma_2)$ *of* A_1, A_2 *such that* $(A_1 \sqcup A_2) = A_i$, i = 1,2.



PROOF. Routine. □

The next Lemma is intended to simplify a verification that some specification indeed specifies a parametrized data type ϕ.

7.2. STANDARD APPLICATION LEMMA.

Suppose that ϕ: ALG(Σ) $\to$ ALG(Δ) *is a persistent parametrized data type. Then the following is a sufficient condition for* $\phi \subseteq (\Gamma,E)^{\Sigma}_{\Delta}$:

$$\textit{for all } A \in \mathrm{Dom}(\phi),\quad \phi(A) = (\Gamma,E)^{\Sigma}_{\Delta}\,(\Sigma,\equiv_A)_{\Sigma}\ .$$

PROOF. $\phi(A) = (\Gamma,E)^{\Sigma}_{\Delta}\,(\Sigma,\equiv_A)_{\Sigma} \Rightarrow$

$$\phi(A) = (\Gamma\ ,\ E \cup \equiv_A)_{\Delta}\ .$$

Let $B = (\Gamma, E\cup\equiv_A)_{\Gamma}$. Then : $B \models E$, $(B)_{\Delta} = \phi(A)$ and $(B)_{\Sigma} = (\phi(A))_{\Sigma} = A$ because of persistency.

Now let $A = (\Sigma',E')_{\Sigma}$, $\Sigma' \cap \Gamma = \Sigma$. We have to show :

$$(\Gamma\cup\Sigma',\ E\cup E')_{\Delta} \doteq \phi(A).$$

Write $A' = (\Sigma',E')_{\Sigma'}$. Then $(A')_{\Sigma} = A = (B)_{\Sigma}$.

Hence, by the Joint Expansion Lemma 7.1, A' and B have a joint expansion $C = A' \sqcup B$ in ALG($\Sigma'\cup\Gamma$) with $(C)_{\Sigma'} = A'$ and $(C)_{\Gamma} = B$.
Clearly $C \models E \cup E'$ and $(C)_{\Delta} = \phi(A)$.

It follows that $(E\cup E')^{\circ} \cap \mathrm{Ter}(\Delta) \subseteq \equiv_{\phi(A)}$. (Def.$^{\circ}$: see proof of Lemma 5.1). Because $(\Gamma\cup\Sigma, E\cup\equiv_A)_{\Delta} = \phi(A)$ we have $(E\cup\equiv_A)^{\circ} \supseteq \equiv_{\phi(A)}$.

Further, $(\Sigma',E')_{\Sigma} = A$ implies $E'^{\circ} \supseteq \equiv_A$. It follows that $(E\cup E')^{\circ} \cap \mathrm{Ter}(\Delta) \supseteq (E\cup\equiv_A)^{\circ} \cap \mathrm{Ter}(\Delta) \supseteq \equiv_{\phi(A)} \cap \mathrm{Ter}(\Delta) = \equiv_{\phi(A)}$.

So $\phi(A) = (\Gamma\cup\Sigma', (E\cup E')^{\circ} \cap \mathrm{Ter}(\Delta))_{\Delta} = (\Gamma\cup\Sigma', E\cup E')_{\Delta}$. □

7.3. DEFINITION. Let (Γ',E') and (Γ,E) be two specifications. We say that (Γ',E') is a *lifting* of (Γ,E), notation: $(\Gamma',E') \trianglerighteq (\Gamma,E)$, if the following three conditions are satisfied:

(i) $\Gamma' \supseteq \Gamma$,

(ii) $\overline{E'} \supseteq E$ ($\overline{\ \ }$ denotes the closure under logical derivability),

(iii) each $A \in \mathrm{ALG}(\Gamma,E)$ can be expanded to an algebra $A' \in \mathrm{ALG}(\Gamma',E')$. (I.e $(A')_{\Gamma} = A$.)

The important property of liftings is the following.

7.4. LIFTING LEMMA.

Let ϕ: ALG(Σ) $\to$ ALG(Δ) *be a persistent parametrized data type. Let* $\Sigma \subseteq \Delta \subseteq \Gamma$ *and assume* $(\Gamma',E') \trianglerighteq (\Gamma,E)$. *Then* :

$$\phi \subseteq (\Gamma,E)^{\Sigma}_{\Delta} \quad \Rightarrow \quad \phi \subseteq (\Gamma',E')^{\Sigma}_{\Delta} . \qquad (*)$$

PROOF. (First note that the requirement that ϕ is persistent, turns the statement (*) into one weaker than the statement $(\Gamma,E)^{\Sigma}_{\Delta} \subseteq (\Gamma',E')^{\Sigma}_{\Delta}$.)

Suppose $A \in \mathrm{Dom}(\phi)$. By Lemma 7.2 it suffices to check that $(\Gamma',E')^{\Sigma}_{\Delta}(\Sigma,\equiv_A) = (\Gamma,E)^{\Sigma}_{\Delta}(\Sigma,\equiv_A)$.

Let $\mathcal{B} = (\Gamma,E \cup \equiv_A)_{\Gamma}$ and $\mathcal{B}' = (\Gamma',E' \cup \equiv_A)_{\Gamma'}$. Take $\mathcal{B}''$ to be an expansion of $\mathcal{B}$ in ALG(Γ') with $\mathcal{B}'' \models E'$. Because of the initiality of $\mathcal{B}'$ there is a homomorphism $\alpha: \mathcal{B}' \longrightarrow \mathcal{B}''$. Restricting α to Γ one obtains $\alpha_{\Gamma} : (\mathcal{B}')_{\Gamma} \longrightarrow (\mathcal{B}'')_{\Gamma} (=\mathcal{B})$. Because $\mathcal{B}' \models E' \cup \equiv_A$, $(\mathcal{B}')_{\Gamma} \models E \cup \equiv_A$. Since $\mathcal{B}$ is initial in ALG(Γ,$E \cup \equiv_A$) , there is a homomorphism $\beta : \mathcal{B} \longrightarrow (\mathcal{B}')_{\Gamma}$. Consequently $\mathcal{B} \cong (\mathcal{B}')_{\Gamma}$ and $(\mathcal{B})_{\Delta} \cong (\mathcal{B}')_{\Delta}$ which had to be shown. □

REFERENCES

[1] BERGSTRA, J.A. & J.V. TUCKER, *Algebraic specifications of computable and semi-computable data structures*, Mathematical Centre, Department of Computer Science Research Report IW 115, Amsterdam 1979.

[2] BERGSTRA, J.A. & J.V. TUCKER, *A characterisation of computable data types by means of a finite equational specification method*, Proc. 7th ICALP, Springer LNCS Vol. 85, 1980.

[3] BERGSTRA, J.A. & J.V. TUCKER, *Initial and final algebra semantics for data type specifications: two characterisation theorems*, Mathematical Centre, Department of Computer Science Research Report IW 131, Amsterdam 1980.

[4] BURSTALL, R.M. & J.A. GOGUEN, *An informal introduction to specifications using* CLEAR, Lecture notes for the International Summer School on theoretical foundations of programming methodology, Munich 1981.

[5] EHRICH, H.D., *On the theory of specification, implementation and parametrization of abstract data types.* Research Report Dortmund 1978.

[6] EHRIG, H.E., H.-J. KREOWSKI, J.W. THATCHER, E.G. WAGNER & J.B. WRIGHT, *Parameterized data types in algebraic specification languages,* Proc. 7th ICALP, Springer LNCS Vol. 85, 1980.

[7] EHRIG, H., *Algebraic theory of parameterized specifications with requirements,* in Proc. of Int. Conf. on the formalization of programming concepts, Springer LNCS Vol. 107.

[8] GANZINGER, H., *Parameterized specifications: parameter passing and optimizing implementation.* Report TUM-18110. Technische Universität München, August 1981.

[9] KAPHENGST, H. & H. REICHEL, *Algebraische Algorithmentheorie,* VEB Robotron, Dresden WIB, 1971.

[10] LEHMANN, D.J. & M.B. SMYTH, *Data types.* Proc. 18th IEEE Symposium on Foundations of Computing, Providence R.I. November 1977.

[11] ROGERS jr., H., *Theory of recursive functions and effective computability* , McGraw-Hill, 1967.

[12] SCOTT, D.S., *Lambda calculus and recursion theory,* in Proc. Third Scandinavian Logic Conf., Ed. S. Kanger, North Holland Studies in Logic and the Foundations of Mathematics, Vol. 82, 1975.

[13] THATCHER, J.W., E.G. WAGNER & J.B. WRIGHT, *Data type specification: parameterization and the power of specification techniques,* Proc. SIGACT 10th Annual Symp. on Theory of Computing, pp. 119-132, May 1978.

[14] WIRSING, M., *An analysis of semantic models for algebraic specifications,* Lecture notes for the International Summer School on theoretical foundations of programming methodology, Munich 1981.