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

J.A. Bergstra
Department of Computer Science
University of Leiden
Wassenaarseweg 80
2300 RA Leiden, The Netherlands

J.W. Klop
Department of Computer Science
Mathematical Centre
Kruislaan 413
1098 SJ Amsterdam, The Netherlands

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.

INTRODUCTION

The mathematical theory of parametrized data types was initially investigated in ADJ [15], [8], LEHMANN & SMYTH [12], KAPHENGST & REICHEL [11] and EHRICH [7]. Central topics in these studies are specification methods and the correctness problem for specifications and parameter passing mechanisms.

Reading through the growing literature 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 [4] and [5].

A parametrized data type will be a partial functor φ: ALG(Σ) $\rightarrow$ ALG(Δ), for some signatures Σ,Δ with $\Sigma \subseteq \Delta$. Here ALG(Γ) denotes the class of all minimal algebras of

signature Γ. (Remark on terminology: BURSTALL & GOGUEN [6] call $A \in ALG(\Gamma)$ an algebra 'without junk'.)

Further, φ is called persistent if $\varphi(A)$ is an expansion of A for all $A \in Dom(\varphi)$. Apart from the requirement that parameter algebras be minimal these definitions correspond to the original ones in ADJ [15].

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 [15]. 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 $\varphi(A)$. 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* $Dom(\varphi) = ALG(\Sigma,E) \cap SCA(\Sigma)$ *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 [9], [10] and [16].

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. The concepts of signature Σ, Σ-algebra, Σ-term are supposed known. $Ter_s(\Sigma)$ is the set of Σ-terms of sort $s \in \Sigma$. 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 $\tau_1 = \tau_1' \wedge \ldots \wedge \tau_k = \tau_k' \rightarrow \tau = \tau'$ 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 domains A_s 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 $A \cong T(\Sigma)/\equiv_A$.

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 [13]) for recursively enumerable (r.e.) subsets of ω; $z \in \omega$ is called an r.e. -index for $V = W_z$.

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

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

Let $\lceil\ \rceil : TER^C(\Sigma) \times Ter^C(\Sigma) \rightarrow \omega$ be a bijective coding of all closed Σ-equations, with $\lfloor\ \rfloor$ as decoding function. Now an arbitrary $\lfloor W_z \rfloor$ need not yet be a congruence; it is after closure under logical derivability: $\overline{\lfloor W_z \rfloor}$.
Coding again it is not hard to see that $\lceil \overline{\lfloor W_z \rfloor} \rceil = W_{c(z)}$ for some recursive $c : \omega \rightarrow \omega$. 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

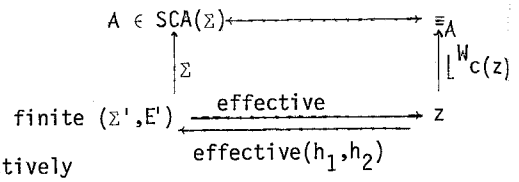
brevity we will use the notation: $(\Sigma',E')_\Sigma = A$. To employ in diagrams, we use the alternative notation: $(\Sigma',E') \xrightarrow{\Sigma} A$.

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

This is proved in BERGSTRA & TUCKER [3]. In fact it is proved there that from an r.e.-index z for $\lceil \equiv_A \rceil$ 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 $\equiv_A$ ($= \lfloor W_z \rfloor$) after coding. Indeed, the following diagram asserts that both kinds of indices can effectively be translated into each other:

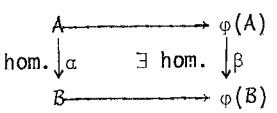


2. PARAMETRIZED DATA TYPES, DESCRIPTIONS AND SPECIFICATIONS

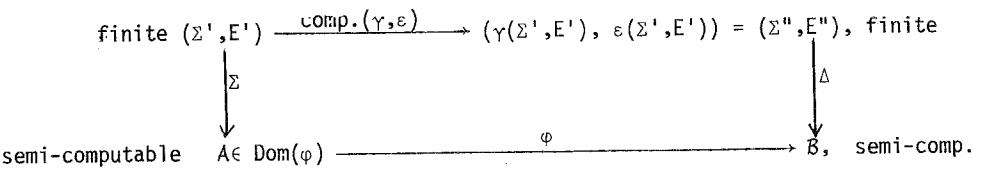
In this section we explain our definition of a parametrized data type, and explain what it means for a parametrized data type to be: effectively given, algebraically specified, continuous or effectively continuous.

2.1. A parametrized data type is a partial functor φ: $ALG(\Sigma) \to ALG(\Delta)$ where $\Sigma \subseteq \Delta$, which satisfies the following condition: for each $A \in Dom(\varphi)$ there is a surjective homomorphism α: $A \to \varphi(A)|_\Sigma$.

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



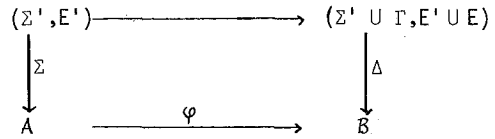
2.2. φ is effectively given (φ is effective) if $Dom(\varphi) \subseteq SCA(\Sigma)$ and there is a pair (γ,ε) of computable operations, acting on finite specifications, that produces a specification $(\gamma(\Sigma',E'), \varepsilon(\Sigma',E'))$ of $\varphi(A)$ for each specification (Σ',E') of some $A \in Dom(\varphi)$.



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

2.3. φ has an algebraic specification if there is a specification (Γ,E) such that for

all $A \in Dom(\varphi)$ this diagram commutes:



If (Γ,E) is finite, then φ has a finite algebraic specification; in that case $\varphi \lceil SCA(\Sigma)$ 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: $\varphi \subseteq (\Gamma,E)^{\Sigma}_{\Delta}$; so the diagram states: $(\Gamma,E)^{\Sigma}_{\Delta}\,(\Sigma',E')_{\Sigma} = (\Sigma' \cup \Gamma, E' \cup E)_{\Delta}$. Note the following composition rule (with $\Gamma' \cap \Gamma = \Delta$): $(\Gamma',F)^{\Delta}_{\Pi} \circ (\Gamma,E)^{\Sigma}_{\Delta} = (\Gamma' \cup \Gamma, F \cup E)^{\Sigma}_{\Pi}$.

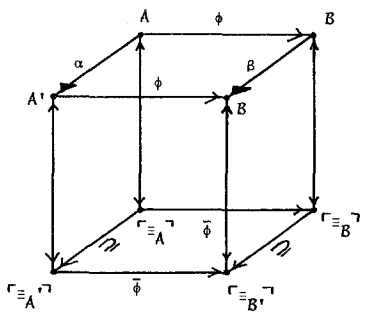
2.4. Representing parametrized data types in reflexive domains

Let $\lceil\ \rceil_\Gamma$ be a bijective coding of closed Γ-equations, and $\lfloor\ \rfloor_\Gamma$ 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

$\lceil Dom(\varphi) \rceil = \{\lceil \equiv_A \rceil^{\Sigma} \mid A \in Dom(\varphi)\}$ and

$\lceil Range(\varphi) \rceil = \{\lceil \equiv_B \rceil^{\Delta} \mid B \in Range(\varphi)\}$.

The mapping $\overline{\varphi} : \lceil Dom(\varphi) \rceil \to \lceil Range(\varphi) \rceil$ is introduced by $\overline{\varphi}(\lceil \equiv_A \rceil) = \lceil \equiv_{\varphi(A)} \rceil$. (See diagram.)



A reflexive domain. The Graph model $P\omega$ is the structure consisting of the powerset of ω and an application operator $\cdot$ on it. Application is defined as follows: for $A,B \in P\omega$,

$A \cdot B = \{m \mid \exists\, n \in \omega\ (n,m) \in A \ \&\ D_n \subseteq B\}$ where $(\ ,\ ): \omega \times \omega \to \omega$ 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: $P\omega \to P\omega$ is continuous if for all $X \in P\omega$:

$F(X) = \bigcup\{F(D_n) \mid D_n \subseteq X\}$. For the next Lemma, see SCOTT [12].

2.4.1. LEMMA *Let* F: $P\omega \to P\omega$. *Then:* ~~F *is continuous*~~ $\iff \exists F \in P\omega\ \forall X \in P\omega\ \ F(X) = F \cdot X$.

2.4.2. DEFINITION. (i) The parametrized data type φ is continuous if $\overline{\varphi}$ is the restriction to $\lceil Dom(\varphi) \rceil$ of some continuous mapping F: $P\omega \to P\omega$. (ii) Moreover, φ is called effectively continuous if $\overline{\varphi}$ is the restriction of a continuous F which is represented in $P\omega$ by an r.e. element $F \in P\omega$. (I.e. F is an enumeration operator, in the sense of ROGERS [13].) (iii) Write RE for the set of r.e. subsets of P . Let φ: RE $\to$ RE . Then φ is called effective if for some computable f: $\forall z\ \ \varphi(W_z) = W_{f(z)}$.

We need the following version of the Theorem of Myhill and Shepherdson (see ROGERS [13]), as stated in SCOTT [14]:

2.4.3. THEOREM. *If* Φ: RE $\to$ RE *is effective, then for some r.e. element* F of $P\omega$: $\forall X \in RE\ \ \Phi(X) = F \cdot X$.

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)$\Leftrightarrow$(iii) will have many generalizations; for instance, in BERGSTRA & KLOP [2] the condition that input algebras are minimal is removed. Other specification methods, such as working with requirements (see EHRIG [9]) 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: $ALG(\Sigma) \to ALG(\Delta)$ *be a persistent parametrized data type with* $Dom(\varphi) = ALG(\Sigma,E) \cap SCA(\Sigma)$, *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: $ALG(\Sigma) \to ALG(\Delta)$ *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: $ALG(\Sigma) \to ALG(\Delta)$ *be a persistent parametrized data type. Then the following are equivalent:*

(i) φ *is effectively continuous;*

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

The structure of the proofs is displayed in the diagrams on the following page.

4. PROVING CONTINUITY

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

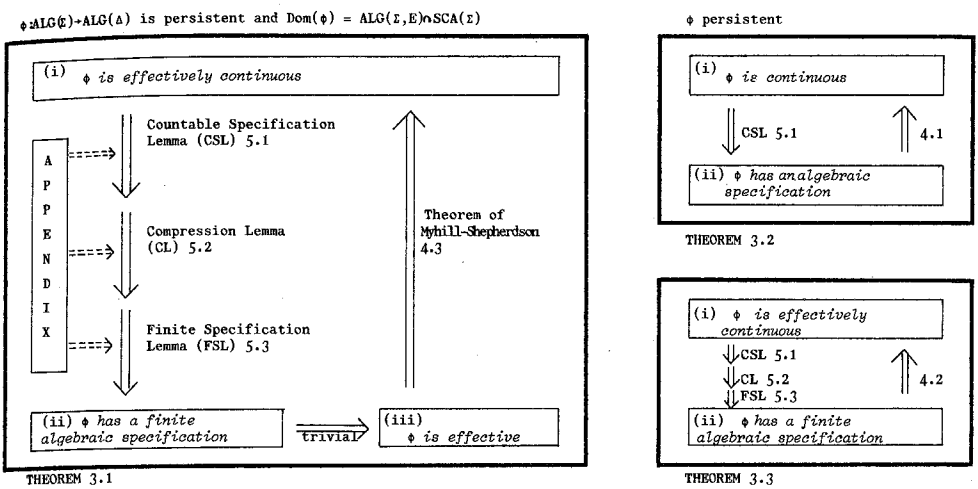
4.1. Proof of Theorem 3.2 (ii) $\Rightarrow$ (i). Let $\lceil\ \rceil$ and $\lfloor\ \rfloor$ be bijective coding and decoding functions for closed Σ-equations, and likewise $[\![\ ]\!]$, $\langle\!|\ |\!\rangle$ for closed Γ-equations.

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

Now let $A = \{(n,m) \mid F \cup \lfloor D_n \rfloor \vdash [\![m]\!]\}$, $A \in P\omega$. Then for $A \in Dom(\varphi)$:

$A \cdot \lceil \equiv_A \rceil = \{m \mid \exists D_n \subseteq \lceil \equiv_A \rceil\ (n,m) \in A\} = \{m \mid \exists D_n \subseteq \lceil \equiv_A \rceil\ F \cup \lfloor D_n \rfloor \vdash [\![m]\!]\} =$
$\{m \mid F \cup \equiv_A \vdash [\![m]\!]\} = \{\lceil e \rceil \mid F \cup \equiv_A \vdash e\} = \{\lceil e \rceil \mid (\Gamma \cup \Sigma, F \cup \equiv_A)_\Delta \models e\} = \lceil \equiv_{\varphi(A)} \rceil = \overline{\varphi}(\lceil \equiv_A \rceil)$.

Hence φ is continuous (by Def. 2.4.2. and Lemma 2.4.1.) □



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) ⇒ (i). Let (γ,ε) be an effective transformation of specifications that describes φ. Consider $\overline{\varphi}$. We will construct an effective operator (see 2.4.2) δ: RE → RE that extends $\overline{\varphi}$. Then it follows by the Theorem of Myhill & Shepherdson (2.4.3) that δ can be extended to an enumeration operator (2.4.2 (ii)), which immediately implies that φ is effectively continuous.

In order to define δ, consider the domain $ALG(\Sigma,E) \cap SCA(\Sigma)$ of φ. Let $W_{d(z)}$ be the coded congruence of an algebra in $ALG(\Sigma,E) \cap SCA(\Sigma)$ 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:

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

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:

$$\delta(W_z) = \{\lceil e \rceil \mid (\gamma(\Sigma'(z),E'(z)), \varepsilon(\Sigma'(z),E'(z))) \vdash e,\ e \text{ is a closed } \Delta\text{-equation}\}$$
$$= W_{g(z)} \quad \text{for an appropriate computable function } g.$$

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

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: $\mathrm{ALG}(\Sigma) \to \mathrm{ALG}(\Delta)$ *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 $\bar{\varphi}$(i.e. F extends $\bar{\varphi}$).
Let $\lceil\ \rceil$, $\lfloor\ \rfloor$, $\lceil\!\lceil\ \rceil\!\rceil$ and $\llbracket\ \rrbracket$ 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

$$e_{(m,n)} = \mathbb{M}\ \lfloor D_m \rfloor \to \llbracket n \rrbracket .$$

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

$(*)$ $\varphi \subseteq (\Delta,E_F)^{\Sigma}_{\Delta}$ 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^0 is the set of all closed equations logically derivable from E

CLAIM. *Let* φ,F *and* E_F *be as above. Then:* (i) $A \in \mathrm{Dom}(\varphi) \Rightarrow (E_F \cup \equiv_A)^0 \subseteq \equiv_{\varphi(A)}$,

(ii) *if* φ *is persistent:*

$A \in \mathrm{Dom}(\varphi) \Rightarrow (E_F \cup \equiv_A)^0 = \equiv_{\varphi(A)}$.

Proof of the claim .

(i) is obvious from the construction of E_F .

(ii) It suffices to show that $\varphi(A) \models E_F \cup \equiv_A$. That $\varphi(A) \models \equiv_A$ is obvious since $(\varphi(A))_\Sigma$ is a homomorphic image of A. Also $\varphi(A) \models E_F$; for, let $e_{(m,n)} \in E_F$. Assume $\varphi(A) \models \mathbb{M}\lfloor D_m \rfloor$. Then also $(\varphi(A))_\Sigma \models \mathbb{M}\lfloor D_m \rfloor$. By persistency $A = (\varphi(A))_\Sigma$, hence $A \models \mathbb{M}\lfloor D_m \rfloor$. Now $A \models \mathbb{M}\lfloor D_m \rfloor \iff \lfloor D_{\mathbb{N}} \rfloor \subseteq \equiv_A \iff D_m \subseteq \lceil \equiv_A \rceil \Rightarrow n \in [\![\equiv_{\varphi(A)}]\!] \iff [\![n]\!] \in \equiv_{\varphi(A)} \iff \varphi(A) \models [\![n]\!]$. Therefore $\varphi(A) \models \mathbb{M}\lfloor D_m \rfloor \to [\![n]\!]$ $(= e_{(m,n)})$, which proves the claim.

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

$$(\Delta, E_F)^\Sigma_\Delta\ (\Sigma, \equiv_A)_\Sigma = (\Delta, E_F \cup \equiv_A)_\Delta = (\Delta, (E_F \cup \equiv_A)^0)_\Delta = \text{(by the claim)}$$

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

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 statement 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:

$$(\Gamma',E') \geq (\Gamma,E) \Rightarrow (\Gamma',E')^\Sigma_\Delta = (\Gamma,E)^\Sigma_\Delta .$$

(In fact we must 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:

$$s_1 = t_1 \to L(\underline{i},\underline{0}) = L(\underline{i},\underline{1}), \ldots, s_{m_i} = t_{m_i} \to L(\underline{i},\underline{m_i - 1}) = L(\underline{i},\underline{m_i})$$

$$L(\underline{i},\underline{0}) = L(\underline{i},\underline{m_i}) \to s'_i = t'_i$$

(Note that using these conditional equations:
$s_1 = t_1 \wedge \ldots \wedge s_{m_i} = t_{m_i} \to L(\underline{i},\underline{0}) = L(\underline{i},\underline{1}) = L(\underline{i},\underline{2}) = \ldots = L(\underline{i},\underline{m_i}) \to s'_i = t'_i$.)

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 $E = \bigcup \{E^{(s,t)} \mid s,t \in \underline{sorts}(\Gamma)\}$ where $E^{(s,t)}$ contains only conditional equations of the form $\tau_1^s = \tau_2^s \to \tau_3^t = \tau_4^t$. Since E is r.e., there are recursive functions $g_i^{(s,t)}$ $(i=1,\ldots,4;\ s,t \in \underline{sorts}(\Gamma))$ such that

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

We define an algebra E as follows. Let Γ^* be a disjoint copy of Γ: for each $s,f,c \in \Gamma$ we have s^*, f^*, $c^* \in \Gamma^*$. We extend $*$ in the obvious way to $Ter(\Gamma)$. Now Σ_E consists of Γ^* augmented by a sort NAT, a constant 0, a function S: NAT $\to$ NAT and for each $s,t \in \underline{sorts}(\Gamma)$ functions $G_i^{(s,t)}$:NAT $\to s^*$ $(i=1,2)$ and $G_i^{(s,t)}$:NAT $\to t^*$ $(i=3,4)$. We write $\underline{k}$ for $S^k(0)$.

E is the minimal algebra specified by the recursive set of closed Σ_E-equations

$$\{G_i^{(s,t)}(\underline{k}) = \lfloor g_i^{(s,t)}(k)\rfloor^* \mid k\in\omega; s,t \in \underline{sorts}(\Gamma)\} .$$

E is computable and, therefore, by Lemma 1.3.1 it has a finite specification (Δ,F).

Now let for each $s \in \underline{sorts}(\Gamma)$ a homomorphism h_s be given satisfying the finite set of equations

$$H = \begin{cases} h^s(c^*) = c \\ h^s(f^*(x_1,\ldots,x_k)) = f(h^{s_1}(x_1),\ldots,h^{s_k}(x_k)) \end{cases}$$

(for all constants c and functions f of Γ).

Finally, define $\mathbb{E} = \{e^{(s,t)} \mid s,t \in \underline{sorts}(\Gamma)\}$ where $e^{(s,t)}$ is the conditional equation $h^s(G_1^{(s,t)}(x)) = h^s(G_2^{(s,t)}(x)) \to h^t(G_3^{(s,t)}(x) = h^t(G_4^{(s,t)}(x))$.

Then, if $(\Gamma',E') = \{\Gamma \cup \Delta \cup \{h^s \mid s \in \underline{sorts}(\Gamma)\}, \mathbb{E} \cup F \cup H\}$, we have $(\Gamma',E') \trianglerighteq (\Gamma,E)$. The routine verification is left to the reader. □

6. PROOF OF THEOREM 3.2 (i) $\Rightarrow$ (ii) AND 3.3 (i) $\Rightarrow$ (ii).

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

The other implication requires some argument. Let φ: ALG(Σ) $\to$ 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 $e \to e'$ only.

Then, using the Finite Specification Lemma (5.3), (Γ,F) is lifted to (Γ',F') with F' finite. By transitivity of lifting, $(\Gamma',F') \trianglerighteq (\Delta,E)$.

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

7. APPENDIX: LIFTINGS OF SPECIFICATIONS (proofs deleted: see [1])

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* $(A_1 \sqcup A_2)_{\Sigma_i} = A_i$, $i = 1,2$.

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: $ALG(\Sigma) \to ALG(\Delta)$ *is a persistent parametrized data type. Then the following is a sufficient condition for* $\varphi \subseteq (\Gamma,E)^{\Sigma}_{\Delta}$: *for all* $A \in Dom(\varphi)$, $\varphi(A) = (\Gamma,E)^{\Sigma}_{\Delta}(\Sigma,\equiv_A)_{\Sigma}$.

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

The important property of liftings is the following.

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

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

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

REFERENCES

[1] BERGSTRA, J.A. & J.W. KLOP, *Algebraic specifications for parametrized data types with minimal parameter and target algebras*, Mathematical Centre, Department of Computer Science Research Report IW 183, Amsterdam 1981.

[2] BERGSTRA, J.A. & J.W. KLOP, *Initial algebra specifications for parametrized data types*, Mathematical Centre, Department of Computer Science Research Report IW 186, Amsterdam 1981.

[3] 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.

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

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

[6] 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.

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

[8] 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.

[9] EHRIG, H., *Algebraic theory of parameterized specifications with requirements*, in Proc. of CAAP81, Springer LNCS, Vol. 112.

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

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

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

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

[14] 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.

[15] 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.

[16] 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.