Theoretical Computer Science 67 (1989) 87-97
North-Holland

ON A CONJECTURE OF BERGSTRA AND TUCKER*

G. MARONGIU
Department of Mathematics, University of Bologna, 40127 Bologna, Italy

S. TULIPANI
Department of Mathematics and Physics, University of Camerino, 62032 Camerino (MC), Italy

Communicated by M. Nivat
Received October 1987

Abstract. Bergstra and Tucker (1983) conjectured that a semicomputable (abstract) data type has a finite hidden enrichment specification under its initial algebra semantics. In a previous paper (1987) we tried to solve the entire conjecture and we found a weak solution. Here, following the line and the proof techniques of the previous paper, we examine a nontrivial case in which the conjecture has a positive answer.

1. Introduction

Since the first pioneering works on the axiomatic approach to data types [12, 15] the need for *finite specifications* has been emphasized. Finite (conditional) equation specifications define, with respect to the initial algebra semantics, abstract data types which are isomorphism classes of semicomputable data structures. Recall that it is common now to think of a data structure as a (multisort) algebra of finite signature, every element of which is represented by a term without variables. The notion of semicomputable algebra is the usual one given by Mal'cev [16] or Rabin [18].

It is easy to find simple semicomputable data types which do not possess any finite specification (see [14]); there are also non-artificial examples (see [5, Section 4] for a discussion and further references). This fact motivated the introduction by the ADJ group [20] of the use of *hidden operations* in algebraic specifications.

In a series of papers [2, 3, 4], Bergstra and Tucker have studied the problem of *specifying absiract data types* which are *computable*, *semicomputable* or *cosemicomputable*. We now briefly recall these notions for future reference (see Section 2 for the formal definition). A recursive number algebra is an algebraic structure based on the natural numbers with recursive operations. An algebra $\mathscr{A}$ is semicomputable, cosemicomputable or computable if it is the quotient of a recursive number algebra by a congruence θ which is r.e., co-r.e. or recursive respectively. The congruence θ will be called the *kernel of the recursive coordinatization of* $\mathscr{A}$ (see Definition 2.1). In particular, Bergstra and Tucker prove the following (see [3, 5]).

* Work performed under the auspices of the Italian MPI and CNR.

Theorem A. *A data structure $\mathcal{A}$ is cosemicomputable iff $\mathcal{A}$ possesses a finite conditional equation specification with hidden functions under final algebra semantics.*

Theorem B. *A data structure $\mathcal{A}$ is computable iff $\mathcal{A}$ possesses two finite conditional equation specifications with hidden functions such that one specifies $\mathcal{A}$ under initial algebra semantics and the other specifies $\mathcal{A}$ under final algebra semantics.*

A specification method which determines a class of (abstract) data types is said to be a *complete method* for that class. A classification of the various known specification methods can be found in [5]. With this terminology the specification methods described in Theorem A and in Theorem B are complete for the classes of cosemicomputable data types and computable data types respectively.

After the previous results it was natural for Bergstra and Tucker to formulate the following conjecture.

Conjecture C. *A data structure $\mathcal{A}$ is semicomputable iff $\mathcal{A}$ possesses a finite conditional specification with hidden functions under initial algebra semantics.*

This conjecture can be formulated in more detail as follows. Given a data structure $\mathcal{A}$ in the finite signature Σ there exist a finite signature Σ' extending Σ with possible new operations but no extra sorts and a finite set of (conditional) equations $\mathscr{E}'$ in Σ' such that the following two conditions are satisfied:

(Con1) Two closed terms t_1, t_2 of signature Σ have the same interpretation in $\mathcal{A}$ if and only if the equation $t_1 = t_2$ can be formally deduced from $\mathscr{E}'$; in symbols,

$$\mathcal{A} \models t_1 = t_2 \text{ iff } \mathscr{E}' \vdash t_1 = t_2;$$

(Con2) For every closed term t' of signature Σ' there exists a closed term t of signature Σ such that the equation $t' = t$ can be formally deduced from $\mathscr{E}'$; in symbols,

$$\mathscr{E}' \vdash t' = t.$$

Trying to solve Conjecture C we proved in [17] that the specification method which requires only condition Con1 is complete for the class of semicomputable data types. In fact, we proved the existence of such an $\mathscr{E}'$ in Σ' and of an extension $\mathcal{A}'$ of $\mathcal{A}$ to signature Σ' such that $\mathcal{A}'$ is a model of $\mathscr{E}'$ and Con1 is true. Our results make use of some techniques relying on the equational theory CL called *Combinatory Logic*. We think that the flexibility of such proof techniques will be useful also for a more complete solution of the conjecture. Although in [17] only the single sort case was treated in detail, the possible extension to the multisorted case was discussed.

In Section 2 of this paper we briefly review the main results of [17] in order to give some idea of the proof techniques employed. Then we shall investigate additional properties of the kernel of recursive coordinatization of the data algebra. In fact, we believe that a complete solution of Conjecture C must pass through a

detailed analysis of the kernel of recursive coordinatization. In [3] Bergstra and Tucker already observe that, as a consequence of their results, the conjecture is true when the data algebra $\mathscr{A}$ has a computable partition. However, this condition is very restrictive beyond the computable case, because there are semicomputable algebras with kernel of recursive coordinatization θ such that no nontrivial union of equivalence classes is recursive. This is the case of equivalences θ that can be defined using selfreferential theories (cf. [9, 21, 6]). In particular, equivalences of this kind are the recursively enumerable precomplete ones (see [21, 6, 7]). We shall show, as the main result of Section 3, that for every data algebra $\mathscr{A}$ whose kernel of recursive coordinatization is recursively enumerable precomplete, Conjecture C is true.

Notation is consistent with the notation in our previous paper [17] and with the current usage in the literature [8, 5] on the algebraic approach to data types. For a detailed discussion of the conceptual relevance of Conjecture C, see [3] and [5].

2. Preliminaries

Let $\Sigma = \{f_1, \ldots, f_k, c_1, \ldots, c_r\}$ be a finite algebraic signature, where $f_1, \ldots, f_k$ are function symbols and $c_1, \ldots, c_r$ are constant symbols. We suppose that Σ has at least one constant symbol. If $\mathscr{A}$ is an algebra of signature Σ and carrier A, we denote by $f_1^A, \ldots, f_k^A, c_1^A, \ldots, c_r^A$ the interpretation of the symbols of Σ in $\mathscr{A}$. A *data structure* is an algebra $\mathscr{A}$ of signature Σ which is generated by the elements $c_1^A, \ldots, c_r^A$; so every element of A is the interpretation of a term without variables in signature Σ. These algebras are also called *minimal* algebras. Moreover, in what follows we assume that every algebra is infinite.

Definition 2.1. An algebra $\mathscr{N}$ of signature Σ is a *recursive number algebra* iff the carrier of $\mathscr{N}$ is the set $\mathbb{N}$ of natural numbers and the operations of $\mathscr{N}$ are recursive functions. A *recursive coordinatization of the algebra* $\mathscr{A}$ is a recursive number algebra $\mathscr{N}$ plus an epimorphism $\pi: \mathscr{N} \to \mathscr{A}$. If $\theta = \ker \pi$ we say that θ is the *kernel of the recursive coordinatization* of $\mathscr{A}$ given by $\mathscr{N}$ and π.

Definition 2.2. A minimal algebra $\mathscr{A}$ is *semicomputable, cosemicomputable* or *computable* iff there exists a recursive coordinatization of $\mathscr{A}$ whose kernel is recursively enumerable, co-recursively enumerable or recursive respectively.

A r.e. equivalence on $\mathbb{N}$ is called a *positive equivalence* in the terminology of Mal'cev [16] and Ershov [9]. A minimal algebra $\mathscr{A}$ will be called *θ-semicomputable* if $\mathscr{A}$ has a recursive coordinatization whose kernel is the positive equivalence θ. Quite analogous definitions can be given for the more general case of multisorted algebras.

In what follows $T(\Sigma)$ will denote the set of terms without variables in the signature Σ and "algebra" will be synonymous with "data structure".

In [17] we obtained the following result.

Theorem 2.3. *Let $\mathscr{A}$ be a semicomputable algebra of signature Σ. Then there exists a finite set $\mathscr{E}^*$ of conditional equations in a signature Σ^* extending Σ (with no extra sorts) such that, for all $t_1, t_2 \in T(\Sigma)$,*

$$\mathscr{A} \models t_1 = t_2 \text{ iff } \mathscr{E}^* \vdash t_1 = t_2.$$

The above theorem proves the existence of a specification $(\Sigma^*, \mathscr{E}^*)$ which satisfies only the condition Con1 mentioned in the Introduction. Such a specification is also called a *consistent extension* for $\mathscr{A}$, while a specification $(\Sigma', \mathscr{E}')$ which satisfies Con1 and Con2 is called a *hidden enrichment* specification under initial algebra semantics (see [8, Chapter 6]).

Owing to the fact that our main result in Section 3 is obtained by a development of the proof of Theorem 2.3, we briefly review the lines along which such a proof was obtained and we recall some useful notions.

Suppose $\mathscr{A}$ is a semicomputable algebra of signature Σ and $(\mathscr{N}, \pi)$ is a recursive coordinatization for which $\theta = \ker \pi$ is positive. Let $m_1, \ldots, m_r$ be natural numbers such that

$$\pi(m_i) = c_i^A, \; m_i = m_j \text{ iff } c_i^A = c_j^A \quad \text{for } i, j = 1, \ldots, r.$$

Let g be a recursive function of three variables such that for $m, n \in \mathbb{N}$

$$(m, n) \in \theta \text{ iff } g(m, n, p) = 0 \quad \text{for some } p \in \mathbb{N}.$$

Denote by cl the signature $\{\bullet, \mathrm{K}, \mathrm{S}\}$ where $\bullet$ is a symbol of binary operation and K, S, are constant symbols. A *combinatory algebra* is a structure $\mathscr{M} = (M, \bullet, \mathrm{K}^M, \mathrm{S}^M)$ of signature cl which satisfies the equational axioms in the variables x, y, z

(E1) $\quad \mathrm{K} \bullet x \bullet y = x,$

(E2) $\quad \mathrm{S} \bullet x \bullet y \bullet z = x \bullet z \bullet (y \bullet z)$

where, as usual, the association for the operation symbol $\bullet$ is to the left. A *combinator* is an element of $T(\mathrm{cl})$, i.e. a term of signature cl with no variables. If n is a natural number we denote by $\ulcorner n \urcorner$ a combinator, called the *numeral* representing n, which is defined inductively by

$$\ulcorner 0 \urcorner = \mathrm{K} \bullet \mathrm{I}, \qquad \ulcorner n+1 \urcorner = \mathrm{S} \bullet \mathrm{B} \bullet \ulcorner n \urcorner \quad \text{for } n \in \mathbb{N}$$

where I, B are the combinators $\mathrm{S} \bullet \mathrm{K} \bullet \mathrm{K}$ and $\mathrm{S} \bullet (\mathrm{K} \bullet \mathrm{S}) \bullet \mathrm{K}$ respectively. CL is the equational theory, named *Combinatory Logic*, whose nonlogical axioms are (E1) and (E2).

If f is an m-ary function symbol in signature Σ, $m > 0$, then let $f^{\mathbb{N}}$ be the interpretation of f in the recursive number algebra $\mathscr{N}$ and let F_f denote a combinator

which in CL represents the recursive function $f^{\mathbb{N}}$. Then, for all natural numbers $n_1, \ldots, n_m$

$$\mathrm{CL} \vdash F_f \bullet \ulcorner n_1 \urcorner \bullet \cdots \bullet \ulcorner n_m \urcorner = \ulcorner f^{\mathbb{N}}(n_1, \ldots, n_m) \urcorner. \tag{2.1}$$

Now let Σ^* be the signature $\Sigma \cup \mathrm{cl} \cup \{\mathrm{Hom}, \mathrm{Nat}\}$, where Hom and Nat are new unary operation symbols. Consider the following list of conditional equations in Σ^* where $x, y, x_1, x_2, \ldots$ denote variables:

(E1)-(E2) axioms of CL,

(E3) $\mathrm{Nat}(\ulcorner 0 \urcorner) = \mathrm{K} \wedge (\mathrm{Nat}(x) = \mathrm{K} \to \mathrm{Nat}(\mathrm{S} \bullet \mathrm{B} \bullet x) = \mathrm{K})$,

(E4) for every f in Σ of arity $m > 0$

$$\bigwedge_{1 \le i \le m} \mathrm{Nat}(x_i) = \mathrm{K} \to \mathrm{Hom}(\mathrm{F}_f \bullet x_1 \bullet x_2 \bullet \cdots \bullet x_m) = f(\mathrm{Hom}(x_1), \ldots, \mathrm{Hom}(x_m)),$$

(E5) $$\bigwedge_{1 \le i \le r} \mathrm{Hom}(\ulcorner m_i \urcorner) = c_i,$$

(E6) $$\mathrm{Nat}(x) = \mathrm{K} \wedge \mathrm{Nat}(y) = \mathrm{K} \wedge \mathrm{Nat}(z) = \mathrm{K} \wedge G \bullet x \bullet y \bullet z = \ulcorner 0 \urcorner \to \mathrm{Hom}(x) = \mathrm{Hom}(y).$$

In axiom (E6) G is a combinator which in CL represents the function g previously introduced that enumerates $\ker \pi$. The set $\mathscr{E}^*$ mentioned in Theorem 2.3 is the set of conditional equations (E1)-(E6).

The idea exploited for writing down the set $\mathscr{E}^*$ can be roughly outlined as follows. The algebra $\mathscr{A}$ can be expanded to a structure $\mathscr{A}^*$ of signature Σ^* whose reduct to the signature cl is a combinatory algebra. Then the structure $\mathscr{N}$ can be codified in the expansion $\mathscr{A}^*$ of $\mathscr{A}$. The unary operation symbol Nat is intended to be interpreted in the "characteristic function" of the subset of the natural numbers of $\mathscr{A}^*$, i.e. the interpretation of the numerals. In this way the structures $\mathscr{N}$ and $\mathscr{A}$ are glued together in the structure $\mathscr{A}^*$ and the unary operation symbol Hom is to be interpreted in the codification of π in $\mathscr{A}^*$. All this is formalized in the following lemma (see [17] for the proof).

Lemma 2.4. *Suppose $\mathscr{A}$ is the algebra of signature Σ previously described. Then, there exists an expansion $\mathscr{A}^*$ of $\mathscr{A}$ to the signature Σ^* which is a model of $\mathscr{E}^*$.*

Now given a term $t \in T(\Sigma)$ we defined a term $\hat{t} \in T(\mathrm{cl})$ which describes the calculation that must be performed in $\mathscr{N}$ to obtain a natural number n such that $\pi(n) = t^{\mathscr{A}}$. The role of term $\hat{t}$ is described in the following lemma.

Lemma 2.5. *For every term $t \in T(\Sigma)$,*

(i) *there exists a natural number n such that $\mathscr{E}^* \vdash \hat{t} = \ulcorner n \urcorner$;*

(ii) $\mathscr{E}^* \vdash \mathrm{Hom}(\hat{t}) = t$.

The proof of Theorem 2.3 now follows from Lemmas 2.4 and 2.5.

Remark 2.6. With the same technique we proved weakened versions of Theorems A and B in [17].

Remark 2.7. The extension of Theorem 2.3 to multisort semicomputable algebras is straightforward. One needs to expand an infinite sort to a combinatory algebra and to add an extra unary function symbol Hom_s for every sort s.

3. Precomplete equivalences

Definition 3.1. A nontrivial equivalence θ on the natural numbers $\mathbb{N}$ is called *precomplete* if for every partial recursive function ψ there exists a total recursive f that makes ψ total modulo θ. This means that for every n in the domain of ψ the pair $(f(n), \psi(n))$ is in θ. An algebra will be called *precomplete* if it is θ-semicomputable for a positive precomplete equivalence θ.

Precomplete equivalences were studied by Ershov [9] and by Visser [21] in connection with some recursion theoretic concepts. The equivalence induced by the provability equivalence of CL on combinators (after gödelization) is a positive precomplete equivalence. This was proved in [21] for the theory $\mathrm{CL}+A_\beta$, but the same proof works for CL. The key point is the existence, in correspondence with every Gödel numbering on combinators, of a special combinator E, called a *universal constructor*, which enumerates all the combinators modulo the theory CL. This means that for every combinator X there exists a natural number n such that

$$\mathrm{CL} \vdash \mathrm{E}\bullet\ulcorner n\urcorner = X. \tag{3.1}$$

Then from the fact that the recursive functions are representable in CL it follows that the equivalence θ defined by

$$(m, n) \in \theta \text{ iff } \mathrm{CL} \vdash \mathrm{E}\bullet\ulcorner m\urcorner = \mathrm{E}\bullet\ulcorner n\urcorner$$

is positive precomplete.

A positive precomplete equivalence is known to be *perfect* (see [9, 7]) which means that any recursive union of equivalence classes must be trivial, i.e. either empty or all of $\mathbb{N}$.

Bergstra and Tucker (see [3, Theorem 4.2]) prove that when a semicomputable minimal algebra $\mathscr{A}$ has a computable partition then it can be finitely specified as a hidden enrichment under initial algebra semantics. However, when $\mathscr{A}$ is θ-semicomputable with θ perfect, no computable partition can exist on A. Moreover, recall that every positive equivalence can be recursively reduced to a positive precomplete equivalence (see [7] for a discussion). So, the case in which the algebra $\mathscr{A}$ has a computable partition is very far from and, in some sense opposite to, the case when $\mathscr{A}$ is precomplete.

Remark. Let α, β be recursively isomorphic equivalences. Then it is immediate to see that a minimal algebra $\mathscr{A}$ is α-semicomputable iff it is β-semicomputable. Recently Lachlan [13] has proved that any two positive precomplete equivalences are recursively isomorphic. Now, if θ is any positive precomplete equivalence then every semicomputable precomplete algebra is in fact θ-semicomputable.

Now, we can state the main result.

Theorem 3.2. *Let $\mathscr{A}$ be a precomplete minimal algebra of signature Σ. Then $\mathscr{A}$ has a finite conditional equation hidden enrichment specification under initial algebra semantics.*

Proof. By the above remark, Theorem 3.2 is proved if we can find some positive precomplete equivalence θ such that every θ-semicomputable algebra $\mathscr{A}$ satisfies the conclusion of Theorem 3.2. To accomplish this let $\mathrm{cln} = \{\bullet, \mathrm{K}, \mathrm{S}, \mathrm{nat}\}$ be the signature obtained from extending the signature cl with a constant symbol nat. Consider the equational theory CLN which extends CL with the following identity in the variable x

(E3′) $\quad \mathrm{nat} \bullet (\mathrm{S} \bullet \mathrm{B} \bullet x) = \mathrm{nat} \bullet x.$

It is possible to prove that a universal constructor can be defined in order to satisfy (3.1) also for terms in a signature which extends cl with a finite number of constants, in particular for signature cln. Such a constructor is a combinator which we still denote by E and which satisfies (3.1) for every X in $T(\mathrm{cln})$. The proof argument is the same as in [1, Corollary 8.1.7].

We will see below (see Proposition 3.4) that CLN is consistent. Therefore, the equivalence θ defined by

$$(m, n) \in \theta \text{ iff } \mathrm{CLN} \vdash \mathrm{E} \bullet \ulcorner m \urcorner = \mathrm{E} \bullet \ulcorner n \urcorner \tag{3.2}$$

is positive precomplete.

Assume now that $\mathscr{N}$ is a recursive number algebra of signature Σ and $\pi : \mathscr{N} \to \mathscr{A}$ an epimorphism such that $\ker \pi = \theta$. Then, we can extend $\mathscr{N}$ and $\mathscr{A}$ to the new signature $\Sigma' = \Sigma \cup \mathrm{cln}$ to obtain

$$\mathscr{N}' = (\mathscr{N}, \mathrm{ap}, m_{\mathrm{K}}, m_{\mathrm{S}}, m_{\mathrm{nat}}), \qquad \mathscr{A}' = (\mathscr{A}, \bullet, \mathrm{K}^{A}, \mathrm{S}^{A}, \mathrm{nat}^{A})$$

such that

ap is a recursive function such that

$$\mathrm{ap}(m, n) = q \text{ iff } \mathrm{CLN} \vdash \mathrm{E} \bullet \ulcorner q \urcorner = (\mathrm{E} \bullet \ulcorner m \urcorner) \bullet (\mathrm{E} \bullet \ulcorner n \urcorner); \tag{3.3}$$

$$\pi(n) = (\mathrm{E} \bullet \ulcorner n \urcorner)^{\mathscr{A}'} \text{ for all } n; \tag{3.4}$$

$$\pi : \mathscr{N}' \to \mathscr{A}' \text{ is a } \Sigma'\text{-epimorphism;} \tag{3.5}$$

$$\mathscr{A}' \text{ restricted to cln is isomorphic to the initial algebra of CLN.} \tag{3.6}$$

As in Section 2 suppose that $m_1, \ldots, m_r$ are natural numbers such that $\pi(m_i) = c_i^A$ for $i = 1, \ldots, r$. Further, for every nonconstant function symbol f in Σ let F_f be a combinator which represents in CL the recursive function which interprets f in the algebra $\mathcal{N}$. Thus, F_f satisfies (2.1).

Now, since $\mathcal{A}$ is a minimal algebra, i.e. generated by $c_1^A, \ldots, c_r^A$, we can choose terms t_K, t_S, t_{nat} in the signature Σ such that

$$t_K^{\mathcal{A}} = K^A, \qquad t_S^{\mathcal{A}} = S^A, \qquad t_{nat}^{\mathcal{A}} = nat^A. \tag{3.7}$$

The following finite set $\mathscr{E}'$ of conditional equations in the signature Σ' specifies the algebra $\mathcal{A}$ as stated in Theorem 3.2.

(E1), (E2), (E3′) axioms of CLN,

(E4′) for every f in Σ of arity $m > 0$

$$\bigwedge_{1 \le i \le m} (nat \bullet x_i = nat \bullet \ulcorner 0 \urcorner) \to E \bullet (F_f \bullet x_1 \bullet \cdots \bullet x_m) = f(E \bullet x_1, \ldots, E \bullet x_m),$$

(E5′) $$\bigwedge_{1 \le i \le r} E \bullet \ulcorner m_i \urcorner = c_i,$$

(E6′) $$t_K = K \wedge t_S = S \wedge t_{nat} = nat.$$

Let us compare the set $\mathscr{E}'$ with the set $\mathscr{E}^*$ considered in Section 2. Axiom (E3) is replaced by axiom (E3′) where, instead of the unary function Nat, we have the constant symbol nat. The role of the two axioms is similar: they axiomatize some properties of the numerals. Axioms (E4′) and (E5′) are obtained from (E4) and (E5) by replacing the function symbol Hom with the universal constructor which codifies the morphism π for (3.2). Thus, axiom (E6) is no longer necessary and is replaced by (E6′) which enables us "to eliminate" the constant symbols of Σ'. In fact condition Con2 allows one to eliminate data introduced by the new symbols of the signature Σ'. To "eliminate" the extra data introduced by the function symbol $\bullet$, the hypothesis that $\mathcal{A}$ is θ-semicomputable is required.

Here is the analogue of Lemma 2.3.

Lemma 3.3. *$\mathcal{A}'$ is a model of $\mathscr{E}'$.*

Proof. Owing to the construction of $\mathcal{A}'$ it is easy to check the truth of (E1), (E2), (E3′), (E5′), (E6′). The truth of (E4′) is established using the following proposition. □

Proposition 3.4. *The theory* CLN *is consistent and for every closed term X in T*(cln) *we have*

$$CLN \vdash nat \bullet X = nat \bullet \ulcorner 0 \urcorner \tag{3.8}$$

iff there exists a natural number n such that $CLN \vdash X = \ulcorner n \urcorner$.

Proof. Add to the weak reduction of CL the new contraction rule: $\mathrm{nat}\bullet(\mathrm{S}\bullet\mathrm{B}\bullet x)\to \mathrm{nat}\bullet x$. Let us denote by $\twoheadrightarrow$ the associated reduction which is the transitive closure of the one-step contraction. With a proof similar to that for the weak reduction (see [19, 11]) it is possible to prove that the reduction $\twoheadrightarrow$ defined above has the Church-Rosser property. Then, CLN is consistent since the equality modulo CLN is determined by $\twoheadrightarrow$.

Now, assume $\mathrm{CLN}\vdash X=\ulcorner n\urcorner$. Then $X\twoheadrightarrow\ulcorner n\urcorner$, because $\ulcorner n\urcorner$ is in normal form. Hence,

$$\mathrm{nat}\bullet X\twoheadrightarrow \mathrm{nat}\bullet\ulcorner n\urcorner\twoheadrightarrow \mathrm{nat}\bullet\ulcorner 0\urcorner,$$

as easily follows.

Vice versa, assume $\mathrm{CLN}\vdash \mathrm{nat}\bullet X=\mathrm{nat}\bullet\ulcorner 0\urcorner$. Then, $\mathrm{nat}\bullet X\twoheadrightarrow\mathrm{nat}\bullet\ulcorner 0\urcorner$, because $\mathrm{nat}\bullet\ulcorner 0\urcorner$ is in normal form. An easy induction on the reduction length shows that there exists a natural number n such that $X\twoheadrightarrow\ulcorner n\urcorner$. Hence, $\mathrm{CLN}\vdash X=\ulcorner n\urcorner$. □

Let us denote by Σ_0 the signature $\Sigma\cup\{\bullet\}$. Then we prove the following lemma.

Lemma 3.5. *For every term $t\in T(\Sigma_0)$ there exist a term $\hat{t}\in T(\mathrm{cl})$ and a natural number n such that*

(i) $\mathscr{E}'\vdash \hat{t}=\ulcorner n\urcorner$;

(ii) $\mathscr{E}'\vdash \mathrm{E}\bullet\hat{t}=t$.

Proof. We define $\hat{t}$ by structural induction as follows:

- If t is c_i for some $i=1,\ldots,r$ then $\hat{t}$ is $\ulcorner m_i\urcorner$;
- if t is $f(t_1,\ldots,t_m)$ for some $f\in\Sigma$ then $\hat{t}$ is $F_f\bullet\hat{t}_1\bullet\cdots\bullet\hat{t}_m$;
- if t is $t_1\bullet t_2$ then $\hat{t}$ is $\mathrm{Ap}\bullet\hat{t}_1\bullet\hat{t}_2$, where Ap is the combinator which represents in CL the recursive function ap of the structure $\mathscr{N}'$.

Now, it is easy to prove (i) and (ii) by induction. □

Lemma 3.6. *Let $t_1, t_2\in T(\Sigma_0)$. Then, $\mathscr{A}'\models t_1=t_2$ implies $\mathscr{E}'\vdash t_1=t_2$.*

Proof. Assume $\mathscr{A}'\models t_1=t_2$. Then from Lemma 3.5 there are n_1, n_2 such that

$$\mathscr{E}'\vdash \mathrm{E}\bullet\ulcorner n_i\urcorner=t_i \quad \text{for } i=1,2. \tag{3.9}$$

Therefore, $\mathscr{A}'\models \mathrm{E}\bullet\ulcorner n_1\urcorner=\mathrm{E}\bullet\ulcorner n_2\urcorner$ and from (3.6) $\mathscr{E}'\vdash \mathrm{E}\bullet\ulcorner n_1\urcorner=\mathrm{E}\bullet\ulcorner n_2\urcorner$. Hence, using (3.9) we finally get $\mathscr{E}'\vdash t_1=t_2$. □

Proof of Theorem 3.2 (*conclusion*). Let $t_1, t_2\in T(\Sigma)$. Then, from Lemmas 3.6 and 3.3 we have

$$\mathscr{A}\models t_1=t_2 \text{ iff } \mathscr{E}'\vdash t_1=t_2.$$

So condition Con1 is satisfied.

We are now able to show that condition Con2 is also satisfied, i.e. the new operation symbols added to Σ do not introduce extra data. If $t' \in T(\Sigma')$, we replace every occurrence of K, S, nat in t' by t_K, t_S, t_{nat} respectively. From axiom (E6′) we get a term $t_0 \in T(\Sigma_0)$ such that $\mathscr{E}' \vdash t' = t_0$. Since the algebra $\mathscr{A}$ is minimal, there exists a term $t \in T(\Sigma)$ such that $\mathscr{A}' \models t = t_0$. Then, from Lemma 3.6 we have $\mathscr{E}' \vdash t = t_0$. Hence, $\mathscr{E}' \vdash t' = t$. □

Remark. In Theorem 2.3 we could also have used a constant symbol nat instead of the unary function symbol Nat and correspondingly the theory CLN instead of CL. Doing so and using axiom (E6′), it is possible to "eliminate" the extra symbols K, S, nat. This means that for every term $t^* \in T(\Sigma^*)$ there exists $t_1 \in T(\Sigma \cup \{\bullet, \mathrm{Hom}\})$ such that $\mathscr{E}^* \vdash t^* = t_1$.

References

[1] H.P. Barendregt, *The Lambda Calculus, its Syntax and Semantics*, Studies in Logic, Vol. 103 (North-Holland, Amsterdam, 1981).

[2] J.A. Bergstra and J.V. Tucker, Algebraic specifications of computable and semicomputable data structures, Dept. of Computer Science Research Report IW 115, Mathematical Centre, Amsterdam, 1979.

[3] J.A. Bergstra and J.V. Tucker, Initial and final algebra semantics for data type specifications: two characterization theorems, *SIAM J. Comput.* **12** (1983) 366–387.

[4] J.A. Bergstra and J.V. Tucker, The completeness of the algebraic specification methods for data types, *Inform. and Control* **54** (1982) 186–200.

[5] J.A. Bergstra and J.V. Tucker, Algebraic specifications for computable and semicomputable data types, Centre for Mathematics and Computer Science Report CS-R8619, Amsterdam, 1986.

[6] C. Bernardi and F. Montagna, Equivalence relations induced by extensional formulae: classification by means of a new fixed point property, *Fund. Math.* **CXXIV** (1984) 221–233.

[7] C. Bernardi and A. Sorbi, Classifying positive equivalence relations, *J. Symb. Logic* **48** (1983) 529–538.

[8] H. Ehrig and B. Mahr, *Fundamentals of Algebraic Specification 1*, EATCS Monograph (Springer, Berlin, 1985).

[9] Ju.L. Ershov, Positive equivalences, *Algebra and Logic* **10** (1973) 378–394.

[10] J.A. Goguen, J.W. Thatcher, E.G. Wagner and J.B. Wright, Abstract data types as initial algebras and correctness of data representations, in: *Proc. ACM Conf. on Computer Graphics, Pattern Recognition and Data Structure*, ACM, New York (1975) 89–93.

[11] J.R. Hindley and J.P. Seldin, *Introduction to Combinators and Lambda-Calculus* (London Math. Soc. Univ. Press, 1986).

[12] C.A.R. Hoare, An axiomatic basis for computer programming, *Comm. ACM* **12** (1969) 576–583.

[13] A.H. Lachlan, A note on positive equivalence relations, *Z. Math. Logik Grundlag. Math.* **33** (1987) 43–46.

[14] M.E. Magister, Limits of the algebraic specification of abstract data types, *ACM Sigplan Notices* **12** (1977) 37–42.

[15] M.E. Magister, Data types, abstract data types and their specification problem, *Theoret. Comput. Sci.* **8** (1979) 89–127.

[16] A.I. Mal'cev, Constructive algebras I, *Russian Math. Surv.* **16** (1961) 77–129.

[17] G. Marongiu and S. Tulipani, Finite algebraic specifications of semicomputable data types, in: *Proc. TAPSOFT '87*, Lecture Notes in Computer Science, Vol. 249 (Springer, Berlin, 1987) 111–122.

[18] M.O. Rabin, Computable algebra, general theory and the theory of computable fields, *Trans. AMS* **98** (1960) 341–360.

[19] L.E. Sanchis, Functionals defined by recursion, *Notre Dame J. Formal Logic* **8** (1967) 161–174.

[20] J.W. Thatcher, E.G. Wagner and J.B. Wright, Data type specifications: parametrization and the power of specification techniques, IBM Research Report RC 7757, Yorktown Heights, NY, 1979.

[21] A. Visser, Numerations, λ-calculus and arithmetic, in: J.R. Hindley and J.P. Seldin, eds., *To H.B. Curry: Essays on Combinatory Logic, Lambda-calculus and Formalism* (Academic Press, New York, 1980) 259–284.