On the Power of Higher-Order Algebraic Specification Methods

P. KOSIUCZENKO
Mathematics Institute, Polytechnic of Warsaw, pl. Politechniki 1, 00-661 Warsaw, Poland

AND

K. MEINKE
Department of Computer Science, University College of Swansea, Swansea SA2 8PP, Great Britain

Soundness and adequacy theorems are presented for the expressive power of higher-order initial algebra specifications with respect to the arithmetical and analytical hierarchies. These results demonstrate that higher-order initial algebra semantics substantially extends the power of both first-order initial and first-order final algebra semantics. It thus provides a unifying framework for all three different approaches to the semantics of algebraic specifications. © 1996 Academic Press, Inc.

INTRODUCTION

The expressive power of first-order algebraic specification methods has been extensively studied over several years and is now largely understood from a theoretical viewpoint. A systematic study of the scope and limits of these methods was undertaken in a series of papers by Bergstra and Tucker (see for example [1–3]). Their work, together with other contributions, is surveyed in [6, 20].

The theory of higher-order algebraic specification is of more recent origin. A survey of this field and its connections to related subjects such as higher-order logic and higher-order term rewriting is the collection of papers [8]. A theoretical study of the power of higher-order algebraic specification methods was initiated in [13], which demonstrated that second-order initial algebra specifications are strictly more powerful than corresponding first-order methods. In this paper we continue our study of the scope and limits of higher-order algebraic specification methods by establishing the first soundness and adequacy results for these methods.

The expressive power of algebraic specification techniques may be characterised by using recursion theory to define various complexity classes of algebras, and by using structural and semantical properties of specifications to define a taxonomy of specification methods. For each taxonomic class \( M \) of specifications one attempts to provide soundness and adequacy theorems. A specification method \( M \) is said to be sound for a complexity class \( C \) if each specification of \( M \) specifies (up to isomorphism) an algebra of complexity \( C \). Conversely, the method \( M \) is said to be adequate for the class \( C \) if every algebra having complexity \( C \) can be specified (up to isomorphism) using a specification of \( M \). A method \( M \) is said to be complete for a complexity class \( C \) if, \( M \) is both sound and adequate for \( C \).

The primary complexity classes used to study first-order algebraic specifications are the classes of computable, semicomputable, and cosemicomputable algebras. These classes of algebras correspond to the lowest levels (respectively \( \Delta^0_1 \), \( \Sigma^0_1 \), and \( \Pi^0_1 \)) obtained using a complexity measure for algebras based on the arithmetical hierarchy. The arithmetical hierarchy provides a classification of the complexity of number theoretic relations based on first-order arithmetical definability and counting quantifier alternations in defining formulas. To characterise the expressive power of higher-order algebraic specifications we require the full arithmetical hierarchy. Indeed, we even require lower levels (up to \( \Pi^1_1 \)) of the analytical hierarchy, an extension of the arithmetical hierarchy based on definability by second-order arithmetical formulas.

Our main results, established in Section 4, confirm the expectation that higher-order algebraic specifications substantially exceed the expressive power of first-order algebraic specifications. This increase in power is already achieved by second-order specification methods. Perhaps surprisingly, our results show that no further increase in expressiveness is achieved by going beyond second-order to higher order methods (at least from a recursion theoretic viewpoint). Obviously, higher-order initial algebra specifications have all the power of first-order initial algebra specifications. Our results show that they also have
all the power of first-order final algebra specifications. Thus higher-order initial algebra semantics unifies all three different approaches to the semantics of algebraic specifications. These conclusions follow from basic results for first-order methods and our two main theorems:

**Soundness Theorem.** Let \( \Sigma \) be a countable higher-order signature and let \( E \) be a recursively enumerable set of \( \Sigma \) equations. The higher-order initial model \( I_{E,1}(\Sigma, E) \) has analytical complexity \( \Pi^1 \).

**Adequacy Theorem.** Let \( \Sigma \) be a countable signature and let \( A \) be a minimal \( \Sigma \) algebra having arithmetical complexity \( \Sigma^\alpha_0 \) or \( \Pi^\alpha_0 \) for some \( \alpha \in \mathbb{N} \). Then \( A \) has a recursive second-order initial model of this specificational algebra. Adequacy Theorem is an encoding of the truth definition for \( \Sigma \)-ary relations.

We prove the Soundness Theorem by a recursion theoretic analysis of the inductive definition of provability in the (infinitary) higher-order equational calculus. This calculus is used to construct the higher-order initial model as a term model. Our main technique in proving the Adequacy Theorem is an encoding of the truth definition for formulas of Peano arithmetic with respect to the standard model \( \mathbb{N} \) of arithmetic using a finite second-order signature and a recursive set of equations. This encoding is presented in Section 3. The second-order initial model of the signature has analytical complexity. For any congruence \( \equiv \) on a recursive number algebra \( R \), if \( \equiv \) has arithmetical complexity then \( \equiv \) is first-order definable by a formula of Peano arithmetic over \( \mathbb{N} \). Thus we can use the encoding of the truth definition for Peano arithmetic in second-order equational logic as hidden machinery to specify every algebra of arithmetical complexity.

The structure of the paper is as follows. In Section 1 we review the basic principles of higher-order algebraic specification. In Section 2 we show how the arithmetical and analytical hierarchies are used to classify the complexity of algebras. In Section 3 we present an encoding of the truth definition for Peano arithmetic in second-order equational logic. Finally, in Section 4, we state and prove our Soundness and Adequacy Theorems.

We have attempted to make the paper largely self-contained. However, we will assume that the reader has some familiarity with the theoretical foundations of first-order algebraic specification methods, a suitable introduction is [4]. Further useful information on the model theory and proof theory of higher-order equations may be found in [11, 12, 14, 16]. A detailed account of the arithmetical and analytical hierarchies can be found in [9] and [18].

### 1. Higher-Order Algebraic Specification

In this section we review the basic principles of higher-order algebraic specification. We begin by making precise our notation for many-sorted universal algebra which is adapted from [15].

We let \( \mathbb{N} \) denote the set of natural numbers, \( P(\mathbb{N}) \) denotes the powerset of \( \mathbb{N} \), \( [\mathbb{N} \to \mathbb{N}] \) denotes the set of all total functions from \( \mathbb{N} \) to \( \mathbb{N} \), and \( B = \{tt, ff\} \) denotes the set of truth values. For any set \( S \), we let \( S^+ \) denote the set of all words or strings over \( S \), including the empty word \( \lambda \). Then \( S^+ \) denotes the set of all non-empty words, \( S^* = S^* - \{\lambda\} \).

#### 1.1. Definition. A many-sorted signature \( \Sigma \) is a pair

\[
\Sigma = (S, \langle \Sigma_{w,s} \mid w \in S^*, s \in S^+ \rangle)
\]

consisting of a non-empty set \( S \), for which each element \( s \in S \) is termed a sort, and an \( S^* \times S^\ast \)-indexed family \( \langle \Sigma_{w,s} \mid w \in S^*, s \in S^+ \rangle \) of sets of constants and operation symbols. For the empty word \( \lambda \in S \) and any sort \( s \in S \), each element \( c \in \Sigma_{w,s} \) is termed a constant symbol of sort \( s \); for each non-empty word \( w = s(1) \cdots s(n) \in S^+ \) and any sort \( s \in S \), each element \( f \in \Sigma_{w,s} \) is termed an operation symbol of domain \( w \), codomain type \( s \), and arity \( n \).

If \( (S_0, \Sigma^0) \) and \( (S_1, \Sigma^1) \) are signatures, we say that \( (S_0, \Sigma^0) \) is a subsignature of \( (S_1, \Sigma^1) \) if, and only if, \( S_0 \subseteq S_1 \) and for each \( w \in S^0 \) and \( s \in S_0 \), we have \( \Sigma^0_{w,s} \subseteq \Sigma^1_{w,s} \).

Let \( \Sigma \) be an \( S \)-sorted signature. An \( S \)-sorted \( \Sigma \) algebra \( A \) is a pair

\[
A = (\langle A_s \mid s \in S \rangle, \langle \Sigma^d_{w,s} \mid w \in S^*, s \in S^+ \rangle)
\]

consisting of an \( S \)-indexed family \( \langle A_s \mid s \in S \rangle \) of sets, the set \( A_s \) being termed the carrier set of sort \( s \) for \( A \), and an \( S^\ast \times S^\ast \) indexed family \( \langle \Sigma^d_{w,s} \mid w \in S^*, s \in S^+ \rangle \) of sets of constants and operations. For each sort \( s \in S \),

\[
\Sigma^d_{w,s} = \langle c_{\bar{s}} \mid c \in \Sigma^d_{w,s} \rangle,
\]

where \( c_{\bar{s}} \in A_s \) is a constant that interprets \( c \) in the algebra \( A \). For each \( w = s(1) \cdots s(n) \in S^+ \) and each \( s \in S \),

\[
\Sigma^d_{w,s} = \langle f_{\bar{s}} \mid f \in \Sigma^d_{w,s} \rangle,
\]

where \( f_{\bar{s}} : A^w \to A_s \) is an operation with domain \( A^w = A_{s(1)} \times \cdots \times A_{s(n)} \), codomain \( A_s \), and arity \( n \) which interprets \( f \) in \( A \).

If \( (S_0, \Sigma^0) \) and \( (S_1, \Sigma^1) \) are signatures, and \( (S_0, \Sigma^0) \) is a subsignature of \( (S_1, \Sigma^1) \) then for any \( \Sigma^1 \) algebra \( A \) there is a unique \( \Sigma^0 \) algebra \( B \), termed the \( \Sigma^0 \) reduct of \( A \), such that for each \( s \in S_0 \), \( B_s = A_s \) and for each \( w \in S^0 \) and \( s \in S_0 \) and each \( f \in \Sigma^0_{w,s} \), \( f_{\bar{s}} = f \). We let \( A |_{\Sigma^0} \) denote the \( \Sigma^0 \) reduct of \( A \).

As usual, we allow \( A \) to denote both a \( \Sigma \) algebra and its \( S \)-indexed family of carrier sets. A \( \Sigma \) algebra \( A \) is said to be minimal (or reachable or term-generated) if, and only if, \( A \) has no proper subalgebra.

Let \( \Sigma \) be an \( S \)-sorted signature and let \( X = \langle X_s \mid s \in S \rangle \) be an \( S \)-indexed family of sets of variable symbols; then
$T(\Sigma, X)$ denotes the term algebra over $\Sigma$ and $X$. We let $T(\Sigma)$ denote the algebra of all closed or ground terms or words over $\Sigma$. If $A$ is a $\Sigma$ algebra and $\alpha = \langle \alpha; X_1 \rightarrow A_1 \mid s \in S \rangle$ is an $S$-indexed family of mappings then $\alpha = \langle \alpha; T(\Sigma, X_1) \rightarrow A_1 \mid s \in S \rangle$ denotes the unique homomorphic extension of $\alpha$, and we also termed the evaluation mapping on terms (under the assignment $\alpha$).

The theory of higher-order universal algebra is developed within the framework of many-sorted first-order universal algebra. We recall the basic definitions of [11] beginning with notations for higher-order types.

1.2. Definition. Let $B$ be any non-empty set, the members of which will be termed basic types, the set $B$ being termed a type basis. The type hierarchy $H(B)$ generated by $B$ is the set $H(B) = \bigcup_{n \in \omega} H_n(B)$ of formal expressions defined inductively by,

$$H_0(B) = B$$

and

$$H_{n+1}(B) = H_n(B) \cup \{ (\sigma \times \tau), (\sigma \rightarrow \tau) \mid \sigma, \tau \in H_n(B) \}.$$  

Each element $(\sigma \times \tau) \in H(B)$ is termed a product type and each element $(\sigma \rightarrow \tau) \in H(B)$ is termed a function type or arrow type.

We can assign an order to each type $\sigma \in H(B)$ as follows. Each basic type $\sigma \in B$ has order 0. If $\sigma, \tau \in H(B)$ have order $m$ and $n$, respectively, then $(\sigma \times \tau)$ has order $\sup\{m, n\}$ and $(\sigma \rightarrow \tau)$ has order $\sup\{m + 1, n\}$.

A type structure $S$ over a type basis $B$ is a subset $S \subseteq H(B)$ which is closed under subtypes in the sense that for any $\sigma, \tau \in H(B)$, if $\sigma \times \tau \in S$ or $(\sigma \rightarrow \tau) \in S$ then both $\sigma \in S$ and $\tau \in S$. We say that $S$ is a basic type structure over $B$ if, and only if, $S \subseteq B$. A type structure $S$ over a basis $B$ is said to be of order $n$ if, and only if, the order of each type $\tau \in S$ is strictly less than $n$. We say that $S$ is an $\omega$-order type structure if, and only if, there is no $n \in \mathbb{N}$ which bounds the order of every type $\tau \in S$.

Given a type structure $S$, a higher-order signature $\Sigma$ is an $S$-sorted signature with distinguished operation symbols for projection and evaluation.

1.3. Definition. Let $S \subseteq H(B)$ be a type structure over a type basis $B$. An $S$-typed signature $\Sigma$ is an $S$-sorted signature such that for each product type $(\sigma \times \tau) \in S$ we have two unary projection operation symbols

$$proj_{\sigma \times \tau}^\sigma \in \Sigma_{(\sigma \times \tau), \sigma}, \quad proj_{\sigma \times \tau}^\tau \in \Sigma_{(\sigma \times \tau), \tau}.$$  

Also for each function type $(\sigma \rightarrow \tau) \in S$ we have a binary evaluation operation symbol

$$eval_{\sigma \rightarrow \tau}^\sigma \in \Sigma_{(\sigma \rightarrow \tau), \sigma, \tau}.$$  

An $S$-typed signature $\Sigma$ is also termed an $n$th-order signature when $S$ is an $n$th-order type structure. When $S$ is a basic type structure then an $S$-typed signature is just an $S$-sorted signature. When the types $\sigma$ and $\tau$ are clear we let $proj^1$ and $proj^2$ denote the projection operation symbols $proj_{\sigma \times \tau}^\sigma$ and $proj_{\sigma \times \tau}^\tau$ respectively and we let $eval$ denote the evaluation operation symbol $eval_{\sigma \rightarrow \tau}^\sigma$.

Next we introduce the intended interpretations of a higher-order signature $\Sigma$.

1.4. Definition. Let $S \subseteq H(B)$ be a type structure over a type basis $B$. Let $\Sigma$ be an $S$-typed signature and let $A$ be an $S$-sorted $\Sigma$ algebra. We say that $A$ is an $S$-typed $\Sigma$ algebra if, and only if, for each product type $(\sigma \times \tau) \in S$ we have $A_{(\sigma \times \tau)} \subseteq A_\sigma \times A_\tau$, and for each function type $(\sigma \rightarrow \tau) \in S$ we have $A_{(\sigma \rightarrow \tau)} \subseteq [A_\sigma \rightarrow A_\tau]$, i.e., $A_{(\sigma \rightarrow \tau)}$ is a subset of the set of all (total) functions from $A_\sigma$ to $A_\tau$. Furthermore, for each product type $(\sigma \times \tau) \in S$ the operations

$$proj^1_{\sigma \times \tau} : A_{(\sigma \times \tau)} \rightarrow A_\sigma, \quad proj^2_{\sigma \times \tau} : A_{(\sigma \times \tau)} \rightarrow A_\tau$$

are the first and second projection operations defined on each

$$a = (a_1, a_2) \in A_{(\sigma \times \tau)}$$

by

$$proj^1_{\sigma \times \tau}(a) = a_1, \quad proj^2_{\sigma \times \tau}(a) = a_2;$$

also, for each function type $(\sigma \rightarrow \tau) \in S$, $eval^\sigma_{\sigma \rightarrow \tau} : A_{(\sigma \rightarrow \tau)} \times A_\sigma \rightarrow A_\tau$ is the evaluation operation on the function space $A_{(\sigma \rightarrow \tau)}$ defined by

$$eval^\sigma_{\sigma \rightarrow \tau}(a, n) = a(n)$$

for each $a \in A_{(\sigma \rightarrow \tau)}$ and $n \in A_\sigma$.

An $S$-typed $\Sigma$ algebra $A$ is also termed an $n$th-order $\Sigma$ algebra when $\Sigma$ is an $n$th-order signature. When $S$ is a basic type structure then an $S$-typed $\Sigma$ algebra is just an $S$-sorted $\Sigma$ algebra.

The structure of an $S$-typed $\Sigma$ algebra can be characterised up to isomorphism by first-order formulas as follows.

1.5. Definition. Let $S \subseteq H(B)$ be a type structure over a type basis $B$, let $\Sigma$ be an $S$-typed signature and let $X$ be an $S$-indexed family of variable sets of variables. The set $Ext = Ext_S$ of extensionality sentences over $\Sigma$ is the set of all $\Sigma$ sentences of the form

$$\forall x \forall y (\forall z (eval_{\sigma \rightarrow \tau}^\sigma(x, z) = eval_{\sigma \rightarrow \tau}^\tau(y, z) \Rightarrow x = y),$$

for each function type $(\sigma \rightarrow \tau) \in S$, where $x, y \in X_{(\sigma \rightarrow \tau)}$, $z \in X_\sigma$, and

$$\forall x \forall y (proj^1(x) = proj^1(y) \land proj^2(x) = proj^2(y) \Rightarrow x = y),$$

for each function type $(\sigma \rightarrow \tau) \in S$, where $x, y \in X_{(\sigma \rightarrow \tau)}$.
for each product type \((\sigma \times \tau) \in S\), where \(x, y \in X_{(\sigma \times \tau)}\). A \(\Sigma\) algebra \(A\) is *extensional* if, and only if, \(A \models \text{Ext}\).

Then we have the following basic representation theorem.

1.6. **Collapsing Theorem** (Mostowski, Shepherdson). Let \(\Sigma\) be an \(S\)-typed signature and let \(A\) be an \(S\)-sorted \(\Sigma\) algebra. Then \(A\) is isomorphic to an \(S\)-typed \(\Sigma\) algebra if, and only if, \(A\) is extensional.

**Proof.** See [11].

For the fundamental principles of equational specification using first-order initial and first-order final algebra semantics we refer the reader to [4] and the survey [20]. We now review the fundamentals of equational specification using higher-order initial algebra semantics.

1.7. **Definition.** Let \(\Sigma\) be an \(S\)-sorted signature and \(X\) be an \(S\)-indexed family of sets of variables. By a *\(\Sigma\) equation* (over \(X\)) we mean a formula of the form

\[ t = t', \]

where for some sort \(s \in S\), \(t, t' \in T(\Sigma, X)\), are terms of sort \(s\) over \(\Sigma\) and \(X\). We say that a \(\Sigma\) equation \(t = t'\) is *ground* if, and only if, \(t\) and \(t'\) contain no variables.

By a *higher-order equational specification* we mean a pair

\[(\Sigma, E)\]

consisting of an \(S\)-typed signature \(\Sigma\) and a set \(E\) of \(\Sigma\) equations.

We let \(\text{Eqn}(\Sigma, X)\) denote the set of all \(\Sigma\) equations. Given any \(\Sigma\) algebra \(A\), we have the usual notion of *truth* for an equation \(e\) under an assignment \(\alpha: X \rightarrow A\), and the usual *validity relation* \(\models\) for an equation \(e\) or set \(E\) of equations with respect to a \(\Sigma\) algebra \(A\) or a class \(K\) of \(\Sigma\) algebras.

The extensional models of a higher-order equational specification \((\Sigma, E)\) form a first-order axiomatisable subclass of the class of all \(\Sigma\) algebras which satisfy \(E\) that is termed an *extensional equational class*.

1.8. **Definition.** Let \(S\) be a type structure over a type basis \(B\). Let \(\Sigma\) be an \(S\)-typed signature, let \(X\) be an \(S\)-indexed family of sets of variables, and let \(E\) be any set of higher-order equations over \(\Sigma\) and \(X\). Define the class \(\text{Alg}_{E,\text{Ext}}(\Sigma, E)\) of all extensional models of \(E\) by

\[ \text{Alg}_{E,\text{Ext}}(\Sigma, E) = \{ A \in \text{Alg}(\Sigma) \mid A \models E \cup \text{Ext} \}. \]

Define the class \(\text{Min}_{E,\text{Ext}}(\Sigma, E)\) of all *minimal extensional models* of \(E\) by

\[ \text{Min}_{E,\text{Ext}}(\Sigma, E) = \{ A \in \text{Alg}(\Sigma) \mid A \models E \cup \text{Ext} \text{ and } A \text{ is minimal} \}. \]

As should be expected, extensional equational classes have weaker closure properties under model theoretic constructions than equational classes. The following theorem generalises well known results from first-order universal algebra.

1.9. **Theorem.** Let \(S\) be a type structure over a type basis \(B\). Let \(\Sigma\) be an \(S\)-typed signature, let \(X\) be an \(S\)-indexed family of sets of variables, and let \(E\) be any set of higher-order equations over \(\Sigma\) and \(X\).

(i) The class \(\text{Alg}_{E,\text{Ext}}(\Sigma, E)\) of all extensional models of \(E\) is closed under the formation of extensional homomorphic images, extensional subalgebras, and direct products.

(ii) The class \(\text{Min}_{E,\text{Ext}}(\Sigma, E)\) contains an initial algebra \(I_{E,\text{Ext}}(\Sigma, E)\).

**Proof.** See [11].

Theorem 1.9 is the starting point for the theory of higher-order algebraic specification. In general, \(\text{Alg}_{E,\text{Ext}}(\Sigma, E)\) does not contain an initial algebra. This is a consequence of the weak closure properties given by part (i) of Theorem 1.9 (which actually characterise extensional equational classes; see [11]). By part (ii), the algebra \(I_{E,\text{Ext}}(\Sigma, E)\) has weaker initiality properties than the usual first-order (non-extensional) initial model \(I(\Sigma, E)\) of \((\Sigma, E)\). We call the algebra \(I_{E,\text{Ext}}(\Sigma, E)\) the *higher-order initial model* of the specification \((\Sigma, E)\). Since \(I_{E,\text{Ext}}(\Sigma, E)\) is extensional, by Theorem 1.6 it is isomorphic to an \(S\)-typed \(\Sigma\) algebra, and thus serves as an appropriate semantics for the specification \((\Sigma, E)\).

The higher-order initial model \(I_{E,\text{Ext}}(\Sigma, E)\) of a specification \((\Sigma, E)\) can be concretely constructed as a term model of \(E\) using an infinitary higher-order equational calculus. This calculus extends the many-sorted first-order equational calculus with additional inference rules for higher types including an *infinitary co-extensional rule*.

1.10. **Definition.** The *infinitary higher-order equational calculus* has the following rules of inference:

(i) For any type \(\tau \in S\) and any term \(t \in T(\Sigma, X)\),

\[ t = t \]

is a *reflexivity* rule.

(ii) For any type \(\tau \in S\) and any terms \(t_0, t_1 \in T(\Sigma, X)\),

\[ t_0 = t_1 \quad t_1 = t_0 \]

is a *symmetry* rule.
(iii) For any type \( \tau \in S \) and any terms \( t_0, t_1, t_2 \in T(\Sigma, X) \),
\[
\frac{t_0 = t_1, \ t_1 = t_2}{t_0 = t_2}
\]
is a transitivity rule.

(iv) For each type \( \sigma \in S \), any terms \( t, t' \in T(\Sigma, X)_\sigma \), any type \( \tau \in S \), any variable symbol \( x \in X \), and any terms \( t_0, t_1 \in T(\Sigma, X) \),
\[
\frac{t = t', \ t_0 = t_1}{t[x/t_0] = t'[x/t_1]}
\]
is a substitution rule.

(v) For each product type \( (\sigma \times \tau) \in S \) and any terms \( t_0, t_1 \in T(\Sigma, X)_{(\sigma \times \tau)} \),
\[
\frac{\text{proj}^1(t_0) = \text{proj}^1(t_1), \ \text{proj}^2(t_0) = \text{proj}^2(t_1)}{t_0 = t_1}
\]
is a projection rule.

(vi) For each function type \( (\sigma \to \tau) \in S \) and any terms \( t_0, t_1 \in T(\Sigma, X)_{(\sigma \to \tau)} \),
\[
\frac{\langle \text{eval}^{(\sigma \to \tau)}(t_0, t) = \text{eval}^{(\sigma \to \tau)}(t_1, t) \mid \ t \in T(\Sigma)_\sigma \rangle}{t_0 = t_1}
\]
is an (infinitary) \( \omega \)-extensionality rule.

Let \( \models_{\omega} \) denote the inference relation between equational theories \( E \subseteq \text{Eqn}(\Sigma, X) \) and equations \( e \in \text{Eqn}(\Sigma, X) \), defined by \( E \models_{\omega} e \) if, and only if, there exists an infinitary proof of \( e \) from \( E \) using the inference rules of the infinitary higher-order equational calculus. To construct the higher-order initial model of a higher-order equational specification \((\Sigma, E)\) as a term model we define the congruence
\[
\equiv_{E, \omega} = \langle \equiv_{E, \omega} \mid \tau \in S \rangle
\]
of provable equivalence (with respect to the infinitary higher-order equational calculus) on the term algebra \( T(\Sigma) \) by
\[
\frac{t \equiv_{E, \omega} t' \iff E \models_{\omega} t = t'}{t \equiv_{E, \omega} t'}
\]
for each type \( \tau \in S \) and any terms \( t, t' \in T(\Sigma)_\tau \).

1.11. THEOREM. Let \((\Sigma, E)\) be a higher-order equational specification. Then
\[
T(\Sigma) \equiv_{E, \omega} I_{E,\omega}(\Sigma, E).
\]


The power of higher-order initial algebra semantics is manifested by the \( \omega \)-extensionality rule and the corresponding strong quotient construction using \( \equiv_{E, \omega} \). It is the complexity of this quotient construction that will be studied in Sections 3 and 4.

By virtue of the construction used for \( T(\Sigma) \equiv_{E, \omega} \) and Theorem 1.11 we have the following completeness result for \( \models_{\omega} \). (For the usual technical reasons, see for example [5] or [15], we impose the assumption of non-voidness on \( \Sigma \); i.e., for each sort \( s \in S \) we assume that there exists a ground term \( t \in T(\Sigma)_s \).

1.12. COMPLETENESS THEOREM. Let \((\Sigma, E)\) be a higher-order equational specification and suppose that \( \Sigma \) is non-void. For any ground equation \( e \in \text{Eqn}(\Sigma, X) \),
\[
E \models_{\omega} e \iff \text{Min}_{E,\omega}(\Sigma, E) \models e.
\]


We conclude this section by making precise the two types of higher-order algebraic specification method that we wish to characterise recursion theoretically.

1.13. DEFINITION. Let \( \text{Spec} = (\Sigma, E) \) be a higher-order equational specification.

(i) Let \( A \) be a \( \Sigma \) algebra. We say that \( \text{Spec} \) specifies \( A \) under higher-order initial algebra semantics if, and only if,
\[
I_{E,\omega}(\text{Spec}) \cong A.
\]

(ii) Let \( \Sigma_0 \) be a subsignature of \( \Sigma \) and let \( A \) be a \( \Sigma_0 \) algebra. We say that \( \text{Spec} \) specifies \( A \) using hidden sorts and hidden operations under higher-order initial algebra semantics if, and only if,
\[
I_{E,\omega}(\text{Spec})|_{\Sigma_0} \cong A.
\]

2. RECURSION THEORETIC COMPLEXITY OF ALGEBRAS

In this section we review the complexity classification for algebras provided by the arithmetical and analytical hierarchies. Further details of this classification may be found in [20]. We begin by recalling the more restricted classification provided by the classes of computable, semi-computable, and cosemicomputable algebras which are taken from the theory of computable algebra (see [10] and [17]).
consisting of a $\Sigma$ recursive number algebra $R$ and an epimorphism $\Theta: R \to A$.

Let $\equiv = \langle \equiv_s | s \in S \rangle$ denote the kernel of $\Theta$ given by

$$r \equiv s \iff \Theta(r) = \Theta(s)$$

for each $s \in S$ and $r, s' \in R$; then

$$A \equiv R/\equiv^0.$$

A $\Sigma$ algebra $A$ is termed computable (respectively semicomputable or cosemicomputable) if, and only if, there exists an effective coordinatization $(R, \Theta)$ of $A$ such that for each sort $s \in S$ the kernel $\equiv_s$ is computable (respectively semicomputable or cosemicomputable).

By basic recursion theory, $A$ is computable if, and only if, $A$ is both semicomputable and cosemicomputable. Furthermore, there exist semicomputable algebras which are not cosemicomputable, and vice-versa. Thus the three complexity classes are distinct. They are sufficient to characterize the scope and limits of all first-order algebra specification methods based on first-order initial semantics ([7]) and first-order final or terminal semantics ([2, 19]). For example, we have the following elementary facts.

2.2. Theorem. Let $S$ be a countable sort set, $\Sigma$ a countable $S$-sorted signature, and $E$ a recursively enumerable set of $\Sigma$ equations:

(i) the initial model $I(\Sigma, E)$ is semicomputable;

(ii) the final model $Z(\Sigma, E)$, if it exists, is cosemicomputable.

Proof. See for example [20].

Various types of converse result hold, for example:

2.3. Proposition. Let $S$ be a finite sort set, $\Sigma$ a finite $S$-sorted signature, and $A$ a minimal $\Sigma$ algebra:

(i) if $A$ is semicomputable then there exists a recursively enumerable set $E_i$ of ground $\Sigma$ equations such that $(\Sigma, E_i)$ specifies $A$ under first-order initial algebra semantics.

(ii) if $A$ is cosemicomputable then there exists a recursively enumerable set $E_i$ of ground $\Sigma$ inequations such that $(\Sigma, E_i)$ specifies $A$ under first-order final algebra semantics.

Proof. See [20].

Thus first-order initial semantics and first-order final semantics provide different and incomparable expressive power. This observation is important since, as we will later show, higher-order initial semantics provides the expressive power of both methods and thus serves as a unifying semantics for all three algebra specification methods.

To characterise the power of higher-order initial algebra specifications we need to extend the classification system of computable, semicomputable, and cosemicomputable algebras. In fact these classes occur at the first three levels in a complexity classification based on the arithmetical hierarchy.

2.4. Definition. The class of arithmetical relations is the smallest class of relations which contains all recursive relations $r \subseteq \mathbb{N}^k \times [\mathbb{N} \to \mathbb{N}]^k$, for all $j, k \in \mathbb{N}$, and is closed under existential ($\exists^0$) and universal ($\forall^0$) number quantification. (The definition of a recursive relation $r \subseteq \mathbb{N}^j \times [\mathbb{N} \to \mathbb{N}]^k$ can be given using oracle computations. We refer the reader to [9].)

The arithmetical hierarchy is the set of classes $\Sigma^0_n$, $\Pi^0_n$, and $\Delta^0_n$ defined by induction on $n$.

(i) $\Sigma^0_0 = \Pi^0_0$ is the class of all recursive relations $r \subseteq \mathbb{N}^j \times [\mathbb{N} \to \mathbb{N}]^k$.

(ii) $\Pi^0_{n+1} = \{ \exists^0 r \mid r \in \Pi^0_n \}$.

(iii) $\Pi^0_{n+1} = \{ \exists^0 r \mid r \in \Sigma^0_n \}$.

(iv) $\Delta^0_n = \Sigma^0_n \cap \Pi^0_n$.

We say that a relation $r \subseteq \mathbb{N}^j \times [\mathbb{N} \to \mathbb{N}]^k$ is $\Sigma^0_n$ (respectively $\Pi^0_n$, $\Delta^0_n$) if, and only if, $r \in \Sigma^0_n$ (respectively $r \in \Pi^0_n$, $r \in \Delta^0_n$).

Let $S$ be an $S$-sorted signature and $A$ a $\Sigma$ algebra. We say that $A$ has complexity $\Sigma^0_n$ (respectively $\Pi^0_n$, $\Delta^0_n$) for $n \in \mathbb{N}$ if and only if there exists an effective coordinatization $(R, \Theta)$ of $A$ such that for each sort $s \in S$, the kernel $\equiv_s$ is $\Sigma^0_n$ (respectively $\Pi^0_n$, $\Delta^0_n$). We say that $A$ has arithmetical complexity if and only if there exists $n \in \mathbb{N}$ such that $A$ has complexity $\Sigma^0_n$ or $\Pi^0_n$.

The classes $\Sigma^0_n$ and $\Pi^0_n$ form a proper hierarchy, for we have the:

2.5. Arithmetical Hierarchy Theorem (Kleene, Mostowski). For all $n > 0$:

(i) $\Sigma^0_n \not\subseteq \Delta^0_n$ and $\Pi^0_n \not\subseteq \Delta^0_n$.

(ii) $\Delta^0_{n+1} \not\subseteq \Sigma^0_n \cup \Pi^0_n$.

Proof. See for example [9].

The complexity classes of Definition 2.4 substantially extend those of Definition 2.1 as is clear from the following fact.

2.6. Proposition. For any sort set $S$, any $S$-sorted signature $\Sigma$ and any $\Sigma$ algebra $A$:

(i) $A$ is semicomputable if and only if $A$ has complexity $\Sigma^0_1$;

(ii) $A$ is cosemicomputable if and only if $A$ has complexity $\Pi^0_1$;

(iii) $A$ is computable if and only if $A$ has complexity $\Delta^0_1$.

Proof. Immediate from Definition 2.1 and the definition of the complexity classes $\Sigma^0_1$, $\Pi^0_1$, and $\Delta^0_1$. 

As we will later show, the complexity of algebras given by higher-order initial algebra specifications goes even beyond the arithmetical hierarchy and into low levels of the analytical hierarchy.

2.7. Definition. The class of analytical relations is the smallest class of relations which contains all arithmetical relations \( r \subseteq \mathbb{N}^k \times [\mathbb{N} \rightarrow \mathbb{N}]^k \) for all \( j, k \in \mathbb{N} \), and is closed under existential (\( 3^1 \)) and universal (\( 1^1 \)) function quantification.

The analytical hierarchy is the set of classes \( \Sigma_n^1, \Pi_n^1, \) and \( A_n^1 \) defined by induction on \( n \):

(i) \( \Sigma_n^1 = \Pi_n^1 = \) the class of all arithmetical relations \( r \subseteq \mathbb{N}^j \times [\mathbb{N} \rightarrow \mathbb{N}]^j \) for all \( j, k \in \mathbb{N} \).

(ii) \( \Sigma_{n+1}^1 = \{ \exists^r | r \in \Pi_n^1 \} \).

(iii) \( \Pi_{n+1}^1 = \{ \forall^r | r \in \Sigma_n^1 \} \).

(iv) \( A_{n+1}^1 = \Sigma_n^1 \cap \Pi_n^1 \).

2.8. Analytical Hierarchy Theorem (Kleene, Mostowski). For all \( n > 0 \):

(i) \( \Sigma_n^1 \not\subseteq A_n^1 \) and \( \Pi_n^1 \not\subseteq A_n^1 \).

(ii) \( A_{n+1}^1 \not\subseteq \Sigma_n^1 \cup \Pi_n^1 \).

Proof. See for example [9].

The arithmetical hierarchy can be characterised in terms of definability by first-order formulas of Peano arithmetic and this fact will play an important role in proving the Adequacy Theorem in Section 4.

Let \( \Sigma^PA \) be the usual (single-sorted) signature for Peano arithmetic consisting of one constant symbol 0 for zero, one unary operation symbol succ for the successor operation, two binary operation symbols + and \( \times \) for the addition and multiplication operations, and one binary relation symbol = for equality. We consider a fixed denumerable set \( V = \{ x_0, x_1, \ldots \} \) of variables. We let \( \text{Term}(\Sigma^PA, V) \) denote the set of all terms over \( \Sigma^PA \) and \( V \) defined inductively in the usual way. We let \( \text{Form}(\Sigma^PA, V) \) denote the set of all first-order formulas over \( \Sigma^PA \) and \( V \), which we take to be the smallest class which contains all atomic formulas (equations) \( (t \equiv t') \), for \( t, t' \in \text{Term}(\Sigma^PA, V) \) and which is closed under negation (\( \neg \phi \)), conjunction (\( \phi \land \psi \)) and universal quantification (\( \forall x \phi \)) for \( x \in V \). The formulas (\( \phi \lor \psi \)) (disjunction) and (\( \exists x \phi \)) (existential quantification) are introduced as abbreviations in the usual way. We also introduce the formula (\( x \leq y \)) (inequality) as an abbreviation for (\( \exists z (x + z = y) \)) and the formula (\( \exists x \leq t \phi \)) (bounded existential quantification) as an abbreviation for (\( \exists x ((x \leq t) \land \phi) \)), where \( x \in V \) and \( t \in \text{Term}(\Sigma^PA, V) \). For any formula \( \phi \in \text{Form}(\Sigma^PA, V) \) we let \( \text{fvar}(\phi) \) denote the set of all variables occurring free in \( \phi \). If \( x_1, \ldots, x_k \) are free variables occurring in \( \phi \) we write \( \phi(x_1, \ldots, x_k) \) to indicate this. If \( t_1, \ldots, t_k \in \text{Term}(\Sigma^PA, V) \) are terms we let \( \phi(x_1/t_1, \ldots, x_k/t_k) \) denote the formula obtained by substituting \( t_j \) for \( x_j \) everywhere it occurs free in \( \phi \), for each \( 1 \leq j \leq k \).

We let \( N \) denote the standard model of arithmetic, namely the \( \Sigma^PA \) structure with universe \( \mathbb{N} \) such that 0, succ, +, \( \times \), and = are interpreted by the constant zero, the successor, addition and multiplication operations on \( \mathbb{N} \), and the equality relation on \( \mathbb{N} \), respectively. We let \( \text{Ass}_N \) denote the set of all assignments from \( V \) to \( \mathbb{N} \). In order to encode an assignment \( \alpha \) as a natural number, later in Section 3, we require that \( \alpha \) be almost everywhere zero. Thus we define

\[
\text{Ass}_N = \{ \alpha : V \rightarrow \mathbb{N} | \alpha(x) = 0 \}
\]

for all but finitely many \( x \in V \).

For any \( \alpha \in \text{Ass}_N \), any variable \( x \in V \), and any \( n \in \mathbb{N} \) we let \( \alpha[x/n] \) denote the assignment that agrees with \( \alpha \) everywhere except on \( x \) where \( \alpha[x/n](x) = n \).

We let \( \models \) denote the usual satisfaction relation for \( \Sigma^PA \) formulas. Thus for any formula \( \phi \in \text{Form}(\Sigma^PA, V) \) and any assignment \( \alpha \in \text{Ass}_N \),

\[
(N, \alpha) \models \phi
\]

denotes that \( \phi \) is true in the structure \( N \) under the assignment \( \alpha \), and

\[
N \models \phi
\]

denotes that \( \phi \) is valid in \( N \). For any formula \( \phi(x_1, \ldots, x_k) \) and for any \( m_1, \ldots, m_k \in \mathbb{N} \) we write \( N \models \phi[m_1, \ldots, m_k] \) to denote the fact that \( (N, \alpha) \models \phi \) for any assignment \( \alpha \in \text{Ass}_N \) satisfying \( \alpha(x_j) = m_j \) for \( j = 1, \ldots, k \).

2.9. Definition. For each \( n \in \mathbb{N} \) we define the subclasses \( \Sigma_n^0, \forall_n^0 \subseteq \text{Form}(\Sigma^PA, V) \) inductively as follows:

(i) \( \forall_n^0 = \forall_n^0 \) is the smallest class of \( \Sigma^PA \) formulas which contains all atomic formulas (\( t = t' \)) for \( t, t' \in \text{Term}(\Sigma^PA, V) \) and is closed under negation (\( \neg \phi \)), conjunction (\( \phi \land \psi \)),
and bounded existential quantification ($\exists x \leq \phi t$), where $x \in V$, $t \in \text{Term}(\Sigma_{PA}, V)$ and $x$ does not occur in $t$.

(ii) $\exists_{n+1}^0 = \{ (\exists x \phi) : \phi \in \Psi_n^0 \land x \in V \}$.

(iii) $\forall_{n+1}^0 = \{ (\forall x \phi) : \phi \in \Psi_n^0 \land x \in V \}$.

For any $k, n \in \mathbb{N}$ and any relation $r \subseteq \mathbb{N}^k$, we say that $r$ is $\exists_n^0$ (respectively $\forall_n^0$)-definable (in the standard model $\mathbb{N}$) if, and only if, there exists a formula $\phi(x_1, ..., x_k) \in \exists_n^0$ (respectively $\forall_n^0$) such that for any $m_1, ..., m_k \in \mathbb{N}$,

$$r(m_1, ..., m_k) \iff \mathbb{N} = \phi(m_1, ..., m_k).$$

2.10. Theorem. For all $k, n > 0$, and any relation $r \subseteq \mathbb{N}^k$,

(i) $r$ is $\Sigma_n^0$ if, and only if, $r$ is $\exists_n^0$-definable, and

(ii) $r$ is $\Pi_n^0$ if, and only if, $r$ is $\forall_n^0$-definable,

over the standard model $\mathbb{N}$ of Peano arithmetic.

Proof. See [9].

Similarly the analytical hierarchy may be characterised in terms of definability by second-order formulas over Peano arithmetic, although we will not need to make use of this fact.

3. A SECOND-ORDER SPECIFICATION OF THE TRUTH DEFINITION OF ARITHMETIC

In this section we give a finite second-order signature $\Sigma_{PA^2}$ and a recursive second-order equational specification $E_{PA^2}$ over $\Sigma_{PA^2}$. The second-order initial model $I_{E_{PA^2}}(\Sigma_{PA^2}, E_{PA^2})$ captures the truth definition for $\Sigma_{PA}$ formulas over the standard model $\mathbb{N}$ of arithmetic. A well known property of this truth definition is that its complexity, and hence the complexity of the initial model $I_{E_{PA^2}}(\Sigma_{PA^2}, E_{PA^2})$, is analytical. In Section 4 we apply this specification as hidden machinery in the proof of the Adequacy Theorem. However, the specification ($\Sigma_{PA^2}, E_{PA^2}$) has its own intrinsic theoretical interest as an encoding of first-order arithmetic into equational logic, though we will not pursue this aspect further here.

We begin by defining the second-order signature $\Sigma_{PA^2}$. We assume the existence of a recursive pairing function $\langle \cdot, \cdot \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and recursive projections $\pi_1, \pi_2 : \mathbb{N} \to \mathbb{N}$ satisfying

$$\pi_1(\langle x, y \rangle) = x, \quad \pi_2(\langle x, y \rangle) = y.$$ 

For example, we may define

$$\langle x, y \rangle = 2^{(x+1)} - 1,$$

and $\pi_1$ and $\pi_2$ accordingly.

3.1. Definition. Let $\Sigma_{PA^2}$ be the second-order type structure

$$\Sigma_{PA^2} = \{ \text{nat, bool, (nat \to nat),(nat \to bool)} \}$$

over the type basis $B = \{ \text{nat, bool} \}$. Let $\Sigma_{PA^2}$ be the $\Sigma_{PA^2}$-typed signature with

$$\Sigma_{PA^2} = \{ \emptyset \}, \quad \Sigma_{PA^2} = \{ \text{suc, p}_1, p_2 \}$$

$$\Sigma_{PA^2} = \{ +, \times \}$$

$$\Sigma_{PA^2} = \{ \text{equal} \}$$

$$\Sigma_{PA^2} = \{ \text{true, false} \}$$

$$\Sigma_{PA^2} = \{ \text{if_then_else} \}$$

$$\Sigma_{PA^2} = \{ \text{true, false} \}$$

$$\Sigma_{PA^2} = \{ \text{eval} \}$$

$$\Sigma_{PA^2} = \{ \text{eval} \}$$

$$\Sigma_{PA^2} = \{ \text{eval} \}$$

$$\Sigma_{PA^2} = \{ \text{eval} \}$$

Our aim is to encode $\Sigma_{PA}$ formulas as ground $\Sigma_{PA^2}$ terms in such a way that the second-order initial model $I_{E_{PA^2}}(\Sigma_{PA^2}, E_{PA^2})$ captures the truth definition with respect to the standard model $\mathbb{N}$. For this we shall encode assignments, terms, and formulas of Peano arithmetic as ground terms over $\Sigma_{PA^2}$. First, let us consider the obvious syntactic representation in $\Sigma_{PA^2}$ of the natural numbers themselves. For each natural number $n \in \mathbb{N}$ we define the numeral representation $\langle n \rangle \in T(\Sigma_{PA^2})_n$ inductively by $\langle 0 \rangle = 0$ and $\langle n + 1 \rangle = \text{suc}(\langle n \rangle)$.

Our next step is to syntactically represent an assignment $\pi \in \mathbb{A}_{\Sigma_{PA^2}}$, which is essentially a finite sequence of natural numbers, by the numeral representation of the finite non-zero part of $\pi$ recursively encoded as a single number.

3.2. Definition. The assignment encoding function

$$\langle \cdot \rangle : \mathbb{A}_{\Sigma_{PA^2}} \to T(\Sigma_{PA^2})_n$$
is defined as follows. For any assignment \( \alpha \in \text{Ass}_N \), if \( \alpha \) is not everywhere 0 then let \( lgh(\alpha) \) be the largest \( n \in \mathbb{N} \) such that \( \alpha(x_n) \neq 0 \) otherwise \( lgh(\alpha) = 0 \). We define

\[
\langle \alpha \rangle = \begin{cases} \\
\text{if } lgh(\alpha) = k \text{ and } k \geq 2; \\
\text{otherwise.}
\end{cases}
\]

The idea for the encoding of terms and formulas of Peano arithmetic is the following. We encode each term \( t \) over \( \Sigma_{PA} \) as a ground term \( \bar{t} \) over \( \Sigma_{PA2} \) of type \( (\text{nat} \rightarrow \text{nat}) \) so that for any assignment \( \alpha \in \text{Ass}_N \),

\[
\overline{I_{Ext}(\Sigma_{PA2}, E_{PA2})}|_{\bar{\alpha}} = \overline{\text{eval}}(\bar{t}, \langle \alpha \rangle).
\]

Then a formula \( \phi \) over \( \Sigma_{PA2} \) is encoded as a ground term \( \bar{\phi} \) over \( \Sigma_{PA2} \) of type \( (\text{nat} \rightarrow \text{bool}) \) in such a way that for any assignment \( \alpha \in \text{Ass}_N \),

\[
(N, \alpha)|\phi \leftrightarrow \overline{I_{Ext}(\Sigma_{PA2}, E_{PA2})}|_{\bar{\alpha}} = \overline{\text{eval}}(\bar{\phi}, \langle \alpha \rangle) = \text{true}.
\]

Next we define the encoding for \( \Sigma_{PA} \) terms.

3.3. Definition. The term encoding function

\[
\overline{\cdot}: \text{Term}(\Sigma_{PA}, V) \rightarrow T(\Sigma_{PA2})(\text{nat} \rightarrow \text{nat})
\]

is defined by induction on the complexity of terms as follows.

(i) For any \( i \in \mathbb{N} \) and variable \( x_i \in V \),

\[
\overline{x_i} = \text{var}((suc\, 0)).
\]

(ii) \( \bar{0} = \bar{\bar{0}} \).

(iii) For any terms \( t_1, t_2 \in \text{Term}(\Sigma_{PA}, V) \),

\[
\begin{align*}
\overline{suc\, (t_1)} &= \overline{suc\, (t_2)} \\
\overline{+(t_1, t_2)} &= \overline{+(t_1, t_2)} \\
\overline{\times(t_1, t_2)} &= \overline{\times(t_1, t_2)}.
\end{align*}
\]

Finally we give the encoding of \( \Sigma_{PA} \) formulas.

3.4. Definition. Define the formula encoding function

\[
\overline{\cdot}: \text{Form}(\Sigma_{PA}, V) \rightarrow T(\Sigma_{PA2})(\text{nat} \rightarrow \text{bool})
\]

by induction on the complexity of formulas as follows.

(i) For any terms \( t_1, t_2 \in \text{Term}(\Sigma_{PA}, V) \),

\[
\overline{(t_1 = t_2)} = \overline{\text{equal}(t_1, t_2)}.
\]

(ii) For any formulas \( \phi, \psi \in \text{Form}(\Sigma_{PA}, V) \),

\[
\overline{\neg \phi} = \overline{\neg(\phi)} \quad \overline{(\phi \land \psi)} = \overline{(\phi \land \psi)}.
\]

For any \( i \in \mathbb{N} \) and variable \( x_i \in V \),

\[
\overline{(\forall x_i \phi)} = \overline{(\forall x_i(\text{suc} \, 0, \phi)).
\]

The operations named in \( \Sigma_{PA2} \) have an intended interpretation which we can now specify with the following recursive set of second-order equations.

3.5. Definition. Let \( E_{PA2} \) be the equational theory over \( \Sigma_{PA2} \) and \( X \) consisting of the following equations and equation schemas:

\[
\begin{align*}
p_1([n]) &= \overline{\text{pi}_1(n)}, \\
p_2([n]) &= \overline{\text{pi}_2(n)} \quad (1.a, b)
\end{align*}
\]

for all \( n \in \mathbb{N} \),

\[
\begin{align*}
+ (0, y) &= y, \\
+ (\text{suc} \, (x), y) &= \text{suc} \, (+ (x, y)) \quad (2.a, b)
\end{align*}
\]

\[
\begin{align*}
\times (0, y) &= 0, \\
\times (\text{suc} \, (x), y) &= +(x, \times (x, y)) \quad (3.a, b)
\end{align*}
\]

\[
\begin{align*}
\text{equal}(0, 0) &= \text{true}, \\
\text{equal}(\text{suc} \, (x), 0) &= \text{false} \quad (4.a, b)
\end{align*}
\]

\[
\begin{align*}
\text{equal}(0, \text{suc} \, (y)) &= \text{false}, \\
\text{equal}(\text{suc} \, (x), \text{suc} \, (y)) &= \text{equal}(x, y)
\end{align*}
\]

\[
\neg(\text{true}) = \text{false}, \quad \neg(\text{false}) = \text{true} \quad (5.a, b)
\]

\[
\begin{align*}
\land (\text{true}, \text{false}) &= \text{false}, \\
\land (\text{false}, \text{true}) &= \text{false} \quad (6.a, b)
\end{align*}
\]

\[
\begin{align*}
\land (\text{true}, \text{true}) &= \text{true} \\
\text{if true then } x_i \text{ else } y_i &= x_i, \\
\text{if false then } x_i \text{ else } y_i &= y_i
\end{align*}
\]

\[
\begin{align*}
\text{eval}(\text{equal}(\text{suc} \, (0)), x) &= p_1(p_2(x)) \quad (8)
\end{align*}
\]

for all \( k \in \mathbb{N} \),

\[
\begin{align*}
\overline{\text{eval}}(0, x) &= 0 \quad (9)
\end{align*}
\]
Recall Definition 3.5. We prove the result by induction on the complexity of $\text{Rec}(\Sigma^{PA}, X)$.

\begin{equation}
\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(t) = \text{eval}(t, \langle \alpha \rangle)).
\end{equation}

**Proof.** Consider any $\bar{\alpha} \in \text{Ass}_N$. We prove the result by induction on the complexity of $\Sigma^{PA}$ terms.

- **Basis.** (i) Consider any $i \in N$ and the variable $x_i \in X$. Then

  $$\bar{\alpha}(x_i) = \text{eval}(\text{var}(\text{suc}(0)), \langle \alpha \rangle).$$

Recall Definition 3.5. Now by Eq. (8),

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(\text{var}(\text{suc}(0)), \langle \alpha \rangle) = p_1(p'_1(\langle \alpha \rangle)), $$

and by (1.a), (1.b), and Definition 3.2,

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(\text{var}(\text{suc}(0)), \langle \alpha \rangle) = \bar{\alpha}(x_i).$$

So

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(x_i) = \text{eval}(\bar{\alpha}, \langle \alpha \rangle)).$$

(ii) Consider the constant symbol $0 \in \Sigma^{PA}$. Then

$$\bar{\alpha}(0) = 0$$

and by Definition 3.3, $0 = \bar{0}$. By Eq. (9),

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(0, \langle \alpha \rangle) = 0.$$

Hence

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(0) = \text{eval}(\bar{0}, \langle \alpha \rangle)).$$

**Induction Step.** (iii) Consider for example the term $+(t_1, t_2)$, where $t_1, t_2 \in \text{Term}(\Sigma^{PA}, V)$. Then

$$\bar{\alpha}(t_1) + \bar{\alpha}(t_2) = \bar{\alpha}(t_1, t_2).$$

Now by Definition 3.3, $+(t_1, t_2) = +(t_1, t_2)$. Also by Eq. (11)

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(+(t_1, t_2), \langle \alpha \rangle) = +(\text{eval}(t_1, \langle \alpha \rangle), \text{eval}(t_2, \langle \alpha \rangle)).$$

By the induction hypothesis

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(t_1), \langle \alpha \rangle) = \text{eval}(t_1, \langle \alpha \rangle),$$

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(t_2), \langle \alpha \rangle) = \text{eval}(t_2, \langle \alpha \rangle).$$

So

$$\text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(+(t_1, t_2), \langle \alpha \rangle) = +(\text{eval}(t_1, \langle \alpha \rangle), \text{eval}(t_2, \langle \alpha \rangle)).$$

and

$$\text{E}^{PA2} \vdash (\exists \alpha (\times(t_1, t_2), \langle \alpha \rangle) = \text{eval}(\times(t_1, t_2), \langle \alpha \rangle).$$

The faithfulness of the formula encoding function with respect to the truth definition for $\Sigma^{PA}$ formulas in the standard model $N$ is established in two lemmas.

**Lemma.** For any formula $\phi \in \text{Form}(\Sigma^{PA}, V)$ and any assignment $\bar{\alpha} \in \text{Ass}_N$,

(a) $\langle N, \bar{\alpha} \rangle \models \phi \Rightarrow \text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(\phi), \langle \alpha \rangle) = \text{true}.$

(b) $\langle N, \bar{\alpha} \rangle \not\models \phi \Rightarrow \text{E}^{PA2} \vdash (\exists \alpha \bar{\alpha}(\phi), \langle \alpha \rangle) = \text{false}.$

**Proof.** Consider any assignment $\bar{\alpha} \in \text{Ass}_N$ and recall Definition 3.5. We prove the result by induction on the complexity of $\phi$. 


Basis. (i) Consider any terms $t_1, t_2 \in \text{Term}((\Sigma^{PA}, V))$ and the atomic formula $(t_1 = t_2)$.

(a) By definition

$$\langle \mathbf{N}, \alpha \rangle \models t_1 = t_2$$

$$\Rightarrow \bar{x}(t_1) = \bar{x}(t_2)$$

$$\Rightarrow E^{PA2} \models_{\omega} \text{equal}(\text{eval}(\overline{T_1}, \langle \alpha \rangle), \text{eval}(\overline{T_2}, \langle \alpha \rangle))$$

$$= \text{true}$$

by Lemma 3.6 and Eqs. (4.a) and (4.d)

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\text{eval}(\overline{T_1}, \overline{T_2}), \langle \alpha \rangle) = \text{true}$$

by Eq. (13)

(b) By definition

$$\langle \mathbf{N}, \alpha \rangle \not\models t_1 = t_2$$

$$\Rightarrow \bar{x}(t_1) \neq \bar{x}(t_2)$$

$$\Rightarrow E^{PA2} \models_{\omega} \text{equal}(\bar{x}(t_1), \bar{x}(t_2)) = \text{false}$$

by Eqs. (4.b), (4.c), and (4.d), and induction on the value of $\bar{x}(t_1)$

$$\Rightarrow E^{PA2} \models_{\omega} \text{equal}(\text{eval}(\overline{T_1}, \langle \alpha \rangle), \text{eval}(\overline{T_2}, \langle \alpha \rangle)) = \text{false}$$

by Lemma 3.6

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\text{eval}(\overline{T_1}, \overline{T_2}), \langle \alpha \rangle) = \text{false}$$

by Eq. (13)

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\overline{T_1} \overline{T_2}), \langle \alpha \rangle) = \text{false}$$

by Definition 3.4.

Induction Step. Consider any formulas $\phi, \psi \in \text{Form}((\Sigma^{PA}, V))$.

(ii) Consider the formula $\neg \phi$.

(a) By definition

$$\langle \mathbf{N}, \alpha \rangle \models \neg \phi$$

$$\Rightarrow \langle \mathbf{N}, \alpha \rangle \not\models \phi$$

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\bar{\phi}, \langle \alpha \rangle) = \text{false}$$

by the induction hypothesis (b)

$$\Rightarrow E^{PA2} \models_{\omega} \neg(\text{eval}(\bar{\phi}, \langle \alpha \rangle)) = \text{true}$$

by Eq. (5.b)

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\overline{\neg \phi}, \langle \alpha \rangle) = \text{true}$$

by Eq. (15)

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\overline{\neg \phi}, \langle \alpha \rangle) = \text{true}$$

by Definition 3.4.

(b) The proof that

$$\langle \mathbf{N}, \alpha \rangle \not\models \neg \phi \Rightarrow E^{PA2} \models_{\omega} \text{eval}(\overline{\neg \phi}, \langle \alpha \rangle) = \text{false}$$

is similar to ii(a) using the induction hypothesis (a).

(iii) Consider the formula $(\phi \land \psi)$.

(a) By definition

$$\langle \mathbf{N}, \alpha \rangle \models (\phi \land \psi)$$

$$\Rightarrow (\mathbf{N}, \alpha) \models \phi \text{ and } (\mathbf{N}, \alpha) \models \psi$$

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\bar{\phi}, \langle \alpha \rangle) = \text{true} \text{ and } E^{PA2} \models_{\omega} \text{eval}(\bar{\psi}, \langle \alpha \rangle) = \text{true}$$

by the induction hypothesis (a)

$$\Rightarrow E^{PA2} \models_{\omega} (\text{eval}(\bar{\phi}, \langle \alpha \rangle), \text{eval}(\bar{\psi}, \langle \alpha \rangle)) = \text{true}$$

by Eq. (6.c)

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\bar{\neg \phi}, \langle \alpha \rangle) = \text{true}$$

by Eq. (16)

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\bar{\neg \phi}, \langle \alpha \rangle) = \text{true}$$

by Definition 3.4.

(b) By definition

$$\langle \mathbf{N}, \alpha \rangle \not\models (\phi \land \psi)$$

$$\Rightarrow (\mathbf{N}, \alpha) \not\models \phi \text{ or } (\mathbf{N}, \alpha) \not\models \psi$$

$$\Rightarrow E^{PA2} \models_{\omega} \text{eval}(\bar{\phi}, \langle \alpha \rangle) = \text{false} \text{ or } E^{PA2} \models_{\omega} \text{eval}(\bar{\psi}, \langle \alpha \rangle) = \text{false}$$

by induction hypothesis (b)

$$\Rightarrow E^{PA2} \models_{\omega} (\text{eval}(\bar{\phi}, \langle \alpha \rangle), \text{eval}(\bar{\psi}, \langle \alpha \rangle)) = \text{false}$$
by equations (6.a) and (6.b)

\[ \Rightarrow E^{PA2} \models \omega eval(\forall x, \phi) = false \]

by Eq. (16)

\[ \Rightarrow E^{PA2} \models \omega eval(\forall x, \phi) = false \]

by Definition 3.4.

(iv) Consider any \( i \in \mathbb{N} \), the variable \( x_i \in V \) and the formula \((\forall x, \phi)\). Suppose that

\[ \text{fear}(\phi) = \{ x_i \} \quad \{ x_i, \ldots, x_n \} \quad (1) \]

(a) By definition

\[ (N, x) \models (\forall x, \phi) \]

by (1) above

\[ \Rightarrow (N, x) \models (\forall x, \phi) = false \]

for all \( n \in \mathbb{N} \)

\[ \Rightarrow (N, x) \models (\forall x, \phi) = false \]

by the induction hypothesis (a), which implies that for all \( \beta \in Ass_\infty \)

\[ E^{PA2} \models \omega eval(\forall x, \phi) = false \]

by Eq. (18.a)

\[ \Rightarrow (N, x) \models (\forall x, \phi) = false \]

by Eqs. (17), (8), (1.a), (1.b) and Definition 3.2

(b) By definition

\[ (N, x) \models (\forall x, \phi) = false \]

for some \( n \in \mathbb{N} \)

\[ \Rightarrow (N, x) \models (\forall x, \phi) = false \]

which implies for such \( n \in \mathbb{N} \) by the induction hypothesis (b)

\[ E^{PA2} \models \omega eval(\forall x, \phi) = false \]

by Eqs. (18.b), (19), (7.a), and (7.b) and induction on the length of the numeral \( \langle x[n] \rangle \)

\[ \Rightarrow (N, x) \models (\forall x, \phi) = false \]

by Eqs. (17), (8), (1.a), (1.b) and Definition 3.2

\[ \Rightarrow (N, x) \models (\forall x, \phi) = false \]

by Definition 3.5.

The converse of Lemma 3.7 follows from the existence of a non-trivial minimal extensional model of \( E^{PA2} \).

3.8. Proposition. There exists a minimal extensional \( \Sigma \) algebra \( A^{PA2} \) such that

\[ A^{PA2} \models E^{PA2} \]

and

\[ true_{A^{PA2}} \neq false_{A^{PA2}}. \]

Proof. Define the \( \Sigma^{PA2} \) algebra \( B \) by \( B_{nat} = N \), \( B_{bool} = B \), \( B_{(nat \rightarrow nat)} = [N \rightarrow N] \), and \( B_{(nat \rightarrow bool)} = [N \rightarrow B] \). The constants and operations of \( B \) are defined in the obvious way to satisfy the equations of \( E^{PA2} \). Let \( A^{PA2} \) be the minimal subalgebra of \( B \); then \( A^{PA2} \) has the required properties.

Using Lemma 3.7 and Proposition 3.8 we establish the converse of Lemma 3.7.

3.9. Lemma. For any formula \( \phi \in Form(\Sigma^{PA}, V) \) and any assignment \( \alpha \in Ass_N \),

(a) \[ E^{PA2} \models \omega eval(\phi(\alpha)) = true \Rightarrow (N, x) \models \phi. \]

(b) \[ E^{PA2} \models \omega eval(\phi(\alpha)) = false \Rightarrow (N, x) \not\models \phi. \]
Proof. (a) Suppose \( E^{PA_2} \models \varphi \), \( \text{eval}(\varphi, \langle x \rangle) = true \). Then for \( A^{PA_2} \) given by Proposition 3.8 above, by the Completeness Theorem 1.12,
\[
\text{eval}(\varphi, \langle x \rangle)_{A^{PA_2}} = true_{A^{PA_2}}.
\]
So by Proposition 3.8
\[
\text{eval}(\varphi, \langle x \rangle)_{A^{PA_2}} \neq false_{A^{PA_2}}.
\]
Thus by the Completeness Theorem 1.12,
\[
E^{PA_2} \not\models \varphi \text{eval}(\varphi, \langle x \rangle) = false.
\]
and so by Lemma 3.7.(b), \((N, \alpha) \models \varphi\).
(b) Follows by a similar argument to (i) using Lemma 3.7.(a).

3.10. Corollary. For any formula \( \phi \in \text{Form}(\Sigma^{PA}, V) \),
\[
N \models \varphi \iff E^{PA_2} \models \varphi = \text{true}.
\]
Proof. By definition
\[
N \models \varphi \iff \forall x \in \text{Ass}_N (N, x) \models \varphi
\]
which, by Lemmas 3.7 and 3.9, holds if, and only if, for all \( x \in \text{Ass}_N \)
\[
E^{PA_2} \models \varphi \iff E^{PA_2} \models \varphi = \text{true}
\]
using the \( \omega \)-extensionality rule.

Let us consider the complexity of the second-order initial
model \( I_{E^d}(\Sigma^{PA_2}, E^{PA_2}) \). Recall the complexity of the set of all valid formulas of Peano arithmetic with respect to the standard model \( N \).

3.11. Theorem. Let \( G: \text{Form}(\Sigma^{PA}, V) \to N \) be a recursive Gödel numbering of \( \Sigma^{PA} \) formulas. The set
\[
\text{Th}_{G}(N) = \{ G(\varphi) \mid \varphi \in \text{Form}(\Sigma^{PA}, V) \text{ and } N \models \varphi \}
\]
is not arithmetical.
Proof. See for example [9].

As a consequence we have:

3.12. Theorem. The second-order initial model
\[
I_{E^d}(\Sigma^{PA_2}, E^{PA_2})
\]
does not have arithmetical complexity.
which contradicts the fact that by Theorem 3.11, $\mathcal{T}_E(N)$ is not arithmetical.

Comparing Theorem 3.12 with Theorem 2.2 and recalling Proposition 2.6, it is clear that higher-order initial algebra specifications may substantially exceed the expressive power of first-order initial and first-order final algebra specifications. In Section 4 we shall determine the extent of this additional expressiveness.

### 4. Soundness and Adequacy Results

In this section we characterise the expressive power of higher-order initial algebra specifications.

We begin with a soundness theorem which is established by a recursion theoretic analysis of the inductive definition of provability $\vdash_\alpha$ for the infinitary higher-order equational calculus. (Recall from Theorem 1.11 that this calculus gives a syntactic construction of the higher-order initial model as a term model.) We recall some basic concepts and results from the theory of inductive definability (see for example [9]).

An operator $\Gamma: P(N) \to P(N)$ is monotone if, and only if, for any $X, Y \subseteq N$,

$$X \subseteq Y \implies \Gamma(X) \subseteq \Gamma(Y).$$

For any monotone operator $\Gamma: P(N) \to P(N)$ we let $\bar{\Gamma} \subseteq N$ denote the least fixed point of $\Gamma$,

$$\bar{\Gamma} = \bigcap \{ Z \subseteq N \mid \Gamma(Z) = Z \}.$$  

We measure the recursion theoretic complexity of $\Gamma$ as follows. For each $\alpha: N \to N$ we let

$$Z_\alpha = \{ n \in N \mid \alpha(n) = 0 \}.$$  

Then we define the relation $R_\Gamma \subseteq N \times [N \to N]$ by

$$R_\Gamma(n, \alpha) \iff n \in \Gamma(Z_\alpha),$$

for any $n \in N$ and $\alpha: N \to N$. We say that $\Gamma$ is $\Sigma^i_\alpha$ (respectively $\Pi^i_\alpha$), for $i = 0, 1$ and $n \in N$, if, and only if, the relation $R_\Gamma$ is $\Sigma^i_\alpha$ (respectively $\Pi^i_\alpha$). We say that $\Gamma$ is arithmetical if, and only if, $\Gamma$ is $\Sigma^0_\alpha$ or $\Pi^0_\alpha$ for some $n \in N$. To prove the Soundness Theorem we require the following fact.

#### 4.1. Proposition
Let $\Gamma: P(N) \to P(N)$ be a monotone operator. If $\Gamma$ is arithmetical then the least fixed point $\bar{\Gamma}$ is $\Pi^1_1$.

**Proof.** See [9].

#### 4.2. Soundness Theorem
For any countable type structure $S$ over a type basis $B$ and for any countable $S$-typed signature $\Sigma$ and recursively enumerable equational specification $E$, the higher-order initial model

$$I_{E,\alpha}(\Sigma, E)$$

has complexity $\Pi^1_1$.

**Proof.** Let $G^{\text{type}}: H(B) \to N$ and $G^{\text{term}}: \bigcup_{\tau \in S} T(\Sigma, X)_{\tau} \to N$ be recursive Gödel numberings of types and terms. For any product type $(\sigma \times \tau) \in S$ and any term $t \in T(\Sigma, X)_{(\sigma \times \tau)}$ we let $\text{proj}_1^G(G^{\text{term}}(t))$ denote $G^{\text{term}}(\text{proj}^1(t))$ and $\text{proj}_2^G(G^{\text{term}}(t))$ denote $G^{\text{term}}(\text{proj}^2(t))$. Also for any function type $(\sigma \to \tau)$ and any terms $t \in T(\Sigma, X)_{(\sigma \to \tau)}$ and $t' \in T(\Sigma, X)_{\tau}$ we let $\text{eval}_G(t, t')$ denote $G^{\text{term}}(\text{eval}(t, t'))$.

The following relations can easily be shown to be recursive:

- **Type**: $\tau \equiv N$, where
  $$\text{Type}(\tau) \iff n \text{ is the Gödel number of a type } \tau \in H(B).$$
- **Producttype**, **Arrowtype**: $\equiv N^3$, where
  $$\text{Producttype}(n_1, n_2, n_3) \iff n_1 \text{ is the Gödel number of a product type } (\sigma \times \tau) \in H(B) \text{ and } n_2 \text{ and } n_3 \text{ are the Gödel numbers of } \sigma \text{ and } \tau \text{ respectively},$$
  $$\text{Arrowtype}(n_1, n_2, n_3) \iff n_1 \text{ is the Gödel number of an arrow type } (\sigma \to \tau) \in H(B) \text{ and } n_2 \text{ and } n_3 \text{ are the Gödel numbers of } \sigma \text{ and } \tau \text{ respectively},$$
- **Term**, **Groundterm**, **Var**: $\equiv N^2$, where
  $$\text{Term}(n_1, n_2) \iff n_1 \text{ is the Gödel number of a term } t \text{ of type } \tau \text{ and } \tau \text{ has Gödel number } n_2,$$
  $$\text{Groundterm}(n_1, n_2) \iff n_1 \text{ is the Gödel number of a ground term } t \text{ of type } \tau \text{ and } \tau \text{ has Gödel number } n_2,$$
  $$\text{Var}(n_1, n_2) \iff n_1 \text{ is the Gödel number of a variable of type } \tau \text{ and } \tau \text{ has Gödel number } n_2,$$
- **Sub**: $\equiv N^4$, where
  $$\text{Sub}(n_1, n_2, n_3, n_4) \iff n_1 \text{ is the Gödel number of a term } t_1 \text{ and } n_2 \text{ is the Gödel number of a term } t_2 \text{ of type } \tau \text{ and } n_3 \text{ is the Gödel number of a variable } x \text{ of type } \tau \text{ and } n_4 \text{ is the Gödel number of the term obtained by substituting } t_2 \text{ for } x \text{ in } t_1.$$

We can extend $G^{\text{term}}$ to a Gödel numbering of equations

$$G^{\text{eq}}: \text{Eq}(\Sigma, X) \to N$$

by

$$G^{\text{eq}}(t_1 = t_2) = \langle G^{\text{term}}(t_1), G^{\text{term}}(t_2) \rangle.$$  

(Recall from Section 2 the recursive pairing function $\langle \cdot, \cdot \rangle: N^2 \to N$.) By assumption the relation $\text{Axiom}_E \equiv N$ given by

$$\text{Axiom}_E(n) \iff n \text{ is the Gödel number of an equation } e \in E$$

is recursively enumerable.
Now define the monotone operator \( \Gamma_{E,\omega} : P(\mathbb{N}) \rightarrow P(\mathbb{N}) \), which will encode the provability relation \( E \vdash_\omega \), as follows. For any \( \alpha : \mathbb{N} \rightarrow \mathbb{N} \), letting

\[
Z_\alpha = \{ n \in \mathbb{N} \mid \alpha(n) = 0 \},
\]
define

\[
m \in \Gamma_{E,\omega}(Z_\alpha) \iff \exists k, k_1 \in \mathbb{N} \text{ such that } Type(n) \text{ and } Term(k, n) \text{ and } m = \langle k, k_1 \rangle \text{ or } \langle k_1, k \rangle \text{ or } \langle k_1, k_1 \rangle
\]

(i) \( \alpha(m) = 0 \), or

(ii) (axiom) \( \text{Axiom}(E(m)) \), or

(iii) (reflexivity) \( \exists k, n \in \mathbb{N} \text{ such that } Type(n) \text{ and } Term(k_1, n) \text{ and } Term(k_2, n) \text{ and } m = \langle k_1, k \rangle \text{ and } \alpha(\langle k, k_1 \rangle) = 0 \), or

(iv) (symmetry) \( \exists k_1, k_2, k_3, n \in \mathbb{N} \text{ such that } Type(n) \text{ and } Term(k_1, n) \text{ and } Term(k_2, n) \text{ and } m = \langle k_1, k \rangle \text{ and } \alpha(\langle k, k_1 \rangle) = 0 \), or

(v) (transitivity) \( \exists k_1, k_2, k_3, n \in \mathbb{N} \text{ such that } Type(n) \text{ and } Term(k_1, n) \text{ and } Term(k_2, n) \text{ and } m = \langle k_1, k_2 \rangle \text{ and } \alpha(\langle k, k_1 \rangle) = 0 \), or

(vi) (substitution) \( \exists k_1, k_2, k_3, n \in \mathbb{N} \text{ such that } Type(n) \text{ and } Term(k_1, n) \text{ and } Term(k_2, n) \text{ and } m = \langle k_1, k \rangle \text{ and } \alpha(\langle k, k_1 \rangle) = 0 \), or

(vii) (projection) \( \exists k_1, k_2, n_0, n_1, n_2 \in \mathbb{N} \text{ such that } Type(n_0) \text{ and } Type(n_1) \text{ and } Producttype(n_0, n_1, n_2) \text{ and } Term(k_1, n_1) \text{ and } Term(k_2, n_2) \text{ and } m = \langle k_1, k \rangle \text{ and } \alpha(\langle k, k_1 \rangle) = 0 \), or

(viii) \( (\omega\text{-extensionality}) \exists k_1, k_2, n_0, n_1, n_2 \in \mathbb{N} \text{ such that } Type(n_0) \text{ and } Type(n_1) \text{ and } Arrowtype(n_0, n_1, n_2) \text{ and } Term(k_1, n_0) \text{ and } Term(k_2, n_0) \text{ and } \forall k \in \mathbb{N} \text{ if } \text{Groundterm}(k, n_1) \text{ then } \alpha(\langle \text{eval}_0(k_1, k_1), \text{eval}_0(k_2, k_2) \rangle) = 0 \) and \( m = \langle k_1, k_1 \rangle \).

Clearly \( \Gamma_{E,\omega} \) is monotone by (i) and arithmetical, so by Proposition 4.1 the least fixed point \( T_{E,\omega} \) has complexity \( \Pi_1^1 \). Also by induction on the complexity of proofs in the infinitary higher-order equational calculus, for any type \( \tau \in S \) and any terms \( t, t' \in T(\Sigma, X) \),

\[
E \vdash_\omega t = t' \Rightarrow G^{\omega}(t = t') \in T_{E,\omega}.
\]

By induction on the sequence \( \Gamma_{E,\omega}^\alpha \) for all ordinals \( \alpha < \omega_1 \), where \( \Gamma_{E,\omega}^0 = \emptyset \) and for any ordinal \( 0 < \alpha < \omega_1 \),

\[
\Gamma_{E,\omega}^\alpha = \Gamma_{E,\omega}^\alpha \left( \bigcup_{\beta < \alpha} \Gamma_{E,\omega}^\beta \right).
\]

we have for every ordinal \( \alpha < \omega_1 \),

\[
G^{\omega}(t = t') \in T_{E,\omega} \Rightarrow E \vdash_\omega t = t'.
\]

Now \( T_{E,\omega} = \bigcup_{\alpha < \omega_1} \Gamma_{E,\omega}^\alpha \) and so

\[
G^{\omega}(t = t') \in T_{E,\omega} \Rightarrow E \vdash_\omega t = t'.
\]

Thus \( \Gamma_{E,\omega} \) encodes the provability relation \( E \vdash_\omega \).

Now define the recursive \( \Sigma \) number algebra \( R \) as follows. For each type \( \tau \in S \) define

\[
R_\tau = \{ n \in \mathbb{N} \mid \text{Groundterm}(n, G^{\text{term}}(\tau)) \}.
\]

For each type \( \tau \in S \) and each constant symbol \( c \in \Sigma_{\omega, \tau} \), define

\[
c_R = G^{\text{term}}(c).
\]

For each \( \tau = (1) \cdots (n) \in S^* \), each \( \tau \in S \), each function symbol \( f \in \Sigma_{\omega, \tau} \) and any ground terms \( t_i \in T(\Sigma)_{(i)} \), for \( 1 \leq i \leq n \), define

\[
f_R(G^{\text{term}}(t_1), \ldots, G^{\text{term}}(t_n)) = G^{\text{term}}(f(t_1, \ldots, t_n)).
\]

Since \( G^{\text{term}} \) is a recursive Gödel numbering of terms then \( R \) is a recursive \( \Sigma \) number algebra. Then define for each type \( \tau \in S \) the relation \( \equiv^{E,\omega}_{\tau} \subseteq \mathbb{N}^2 \) by

\[
m \equiv^{E,\omega}_{\tau} n \iff \langle m, n \rangle \in T_{E,\omega} \text{ and } \text{Groundterm}(m, G^{\text{term}}(\tau)) \text{ and } \text{Groundterm}(n, G^{\text{term}}(\tau))
\]

for any \( m, n \in \mathbb{N} \). Then for each type \( \tau \in S \) the relation \( \equiv^{E,\omega}_{\tau} \subseteq \mathbb{N}^2 \) and has complexity \( \Pi_1^1 \) since \( \text{Groundterm} \) is recursive. Also, since \( \Gamma_{E,\omega} \) encodes the provability relation \( E \vdash_\omega \), for any type \( \tau \in S \) and ground terms \( t, t' \in T(\Sigma) \),

\[
E \vdash_\omega t = t' \iff G^{\text{term}}(t) \equiv^{E,\omega}_{\tau} G^{\text{term}}(t').
\]

So the family \( \equiv^{E,\omega}_{\tau} = \{ \equiv^{E,\omega}_{\tau} \mid \tau \in S \} \) is a congruence on \( R \) and

\[
R/\equiv^{E,\omega}_{\tau} \cong \Sigma.
\]

Therefore \( \Lambda \) has complexity \( \Pi_1^1 \).

Observe that Theorem 4.2 concerns arbitrary (even \( \omega \)) order specifications. As a consequence of the following adequacy theorem we observe that no substantial increase in expressiveness (at least from a recursion theoretic viewpoint) occurs when we move from second-order initial algebra specifications, to specifications of order greater than or equal to 3.

By using the recursive second-order equational specification \( (\Sigma^{PA^2}, \cdot^{PA^2}) \) of the truth definition of arithmetic, given in Section 3, as hidden machinery we can give a recursive
second-order equational specification of any countable minimal algebra having arithmetical complexity.

4.3. Adequacy Theorem. For any countable $S$-sorted signature $\Sigma$ and any minimal $\Sigma$ algebra $A$, if $A$ has arithmetical complexity then $A$ has a recursive second-order equational specification with hidden sorts and hidden functions. If $|S|=n$ then two hidden sorts and $27+2n$ hidden functions suffice.

Proof. Suppose that $A$ has arithmetical complexity. Then there exists an effective coordinatization $(R, \theta)$ of $A$ such that for each sort $s \in S$ the kernel $\equiv^\theta_S$ has arithmetical complexity.

Since $R$ is a recursive number algebra then the ground term evaluation mapping

$$Val: T(\Sigma) \rightarrow R$$

is an $S$-indexed family of recursive G"{o}del numberings of terms. Consider the epimorphism $\theta \cdot Val: T(\Sigma) \rightarrow A$ and its associated kernel $\equiv^{\theta \cdot Val}_A$. Then

$$A \cong T(\Sigma)/\equiv^{\theta \cdot Val}_A. \quad (1)$$

Now for each sort $s \in S$, since $\equiv^\theta_S$ is arithmetical, by Theorem 2.10 there exists a $\Sigma_{PA}^2$ formula

$$\phi_{\theta, s}(x, y)$$

with just two free variables $x, y \in V$ such that for any $m, n \in R_s$

$$m \equiv^\theta_S n \Leftrightarrow N \models \phi_{\theta, s}(m, n). \quad (2)$$

We make a recursive second-order equational specification with hidden sorts and hidden functions

$$\Sigma(A), E(A)$$

of $A$ as follows. Define

$$S(A) = S \cup S_{PA}$$

and define the $S(A)$-typed signature $\Sigma(A)$ as follows.

For each sort $s \in S$,

$$\Sigma(A)_{s, x, s} = \Sigma_{s, x} \cup \{\text{identify}\}$$

and

$$\Sigma(A)_{\text{bool}, s, s} = \{\text{if then else}\},$$

for each $w \in S^*$ and $s \in S$ with $w \neq ss$

$$\Sigma(A)_{w, s} = \Sigma_{w, s},$$

for each $w \in S_{PA}^*$ and $s \in S_{PA}$,

$$\Sigma(A)_{w, s} = \Sigma_{w, s}$$

and for all other $w \in S(A)^*$ and $s \in S(A)$, $\Sigma(A)_{w, s} = \emptyset$. Thus $\Sigma(A)$ is the result of joining together $\Sigma$ with $\Sigma_{PA}^2$ and adding two new operation symbols identify and if then else for each sort $s \in S$.

Define $E(A)$ to consist of all equations of $E_{PA2}$ together with, for each sort $s \in S$, the equations and equation schemas

$$\begin{align*}
\text{if true then x else y} &= x & (3) \\
\text{if false then x else y} &= y & (4) \\
\text{identify}(x, y) &= x & (5) \\
\text{identify}(t, t') &= \text{eval}(\phi_{\theta, s}(x, y)| Val_A(t), y| Val_A(t'), 0) & (6)
\end{align*}$$

for $x$ and $y$ distinct variables of sort $s$ and for all ground terms $t, t' \in T(\Sigma)$.

Clearly by joining together the algebras $A$ and $A_{PA2}$ we can define a second-order $\Sigma(A)$ algebra $B$ such that

$$B \models E(A).$$

Since $E_{PA2} \subseteq E(A)$ and $B \models E(A)$ then

$$I_{EPA2}(\Sigma(A), E(A)) \subseteq I_{EPA2}(\Sigma_{PA2}, E_{PA2}). \quad (7)$$

For any sort $s \in S$ and any terms $t, t' \in T(\Sigma)_s$,$$
$$t \equiv^\theta_S t' \Leftrightarrow Val_A(t) \equiv^\theta_S Val_A(t')$$

$$\Leftrightarrow N \models \phi_{\theta, s}(Val_A(t), Val_A(t'))$$

by (2),

$$\Leftrightarrow E(A) \models ev(\phi_{\theta, s}(x)| Val_A(t), y| Val_A(t'), 0) = \text{true}$$

by Lemmas 3.7 and 3.9 since $E_{PA2} \subseteq E(A)$,

$$\Leftrightarrow E(A) \models t = t'$$

for each $w \in S^*$ and $s \in S$ with $w \neq ss$. 

\end{document}
by Eqs. (3), (5), and (6) above and the fact that $B \models E(A)$,

$$
\iff t \equiv E(A), \alpha t'.
$$

(8)

Furthermore, by (7) for every term $t \in T(\Sigma(A))_{\text{bool}}$ either $E(A) \models t = \text{true}$ or $E(A) \models t = \text{false}$. So by Eqs. (3), (4) and (5), for every sort $s \in S$ and every term $t \in T(\Sigma(A))_s$ there exists a term $t' \in T(\Sigma)_s$ such that

$$
E(A) \models t = t'.
$$

(9)

Then by (8), (9), and (1),

$$
A \cong I_{E(A)}(\Sigma(A), E(A)) \models \Sigma.
$$

Obviously, if $|S| = n$ then we have used only four hidden sorts and $27 + 2n$ hidden functions in $\Sigma(A)$. By representing the booleans as natural numbers we can use just two hidden sorts. Further optimisations are possible.

Theorem 4.3 should be contrasted with Theorem 2.2 and Proposition 2.6 to compare the power of higher-order initial algebra specifications with first-order initial and first-order final algebra specifications. It is clear from these results that higher-order initial algebra specifications properly include the expressive power of both first-order initial and first-order final algebra specifications. In this sense they provide a unified approach to the semantics of first-order algebraic specification methods.

Notice that Theorem 4.3 does not establish the converse of Theorem 4.2 but rather a slightly weaker result. The problem of establishing a completeness theorem which exactly characterises the power of higher-order initial algebra specifications remains open.

5. CONCLUSIONS

The results established in this paper demonstrate the substantial power of higher-order initial algebra specifications. Furthermore, they suggest that higher-order initial semantics can provide a unified account of the semantics of first-order algebraic specifications. An important open problem, to be addressed by future research, is to find a suitable completeness theorem which exactly characterises the power of higher-order initial algebra specifications.

This research was partially supported by funds from EU ESPRIT Working Groups 7232 (Gentzen) and 8533 (Nada).

Received July 22, 1994; final manuscript received August 12, 1995

REFERENCES


