SIAM J. COMPUT.
Vol. 30, No. 6, pp. 2084–2096

FIRST-ORDER SPECIFICATIONS OF PROGRAMMABLE DATA TYPES*

GRAŻYNA MIRKOWSKA†, ANDRZEJ SALWICKI†, MARIAN SREBRNY‡, AND ANDRZEJ TARLECKI§

Abstract. We consider first-order specifications together with the restriction to accept only programmable algebras as models. We provide a criterion which links this approach with the "generation principle": all programmable models of any specification SP that meets this criterion are reachable. We also show an example of a specification which does not satisfy the criterion and admits a programmable yet nonreachable model. Moreover, a general method of showing the existence of programmable but nonreachable models for a class of first-order specifications is given.

Key words. algebraic specification, reachable algebra, data types

AMS subject classifications. 68Q60, 68Q65, 68P05

PII. S0097539797322528

1. Introduction. The problem of characterizing intended algebras is important in various areas of computer science. In the areas of specification, development, and validation of software systems one needs to describe the intended models—typically "no junk" algebras—modelling (representing) possible or actual implementations of software systems being specified.

In this paper we make use of intrinsic conceptual parallelism between software engineering and metamathematics. Some software modules—classes, libraries, packets, etc.—can be conveniently viewed as algebraic structures. The goal of specifying software modules then finds its formal counterpart: specification of algebraic structures. In the terminology of computer scientists one sometimes also says specification of data types. A formal specification may (and should) serve as the only source of information on a data structure $\mathcal{A}$ when one constructs and/or analyzes a program P which uses the data structure $\mathcal{A}$. Suppose that such a data structure $\mathcal{A}$ is described by a specification SP. Is the specification SP sufficiently rich in order to ensure that any proof of a valid semantical property of the program P (such as correctness, termination, etc.) can rely only on the axioms listed in SP? Or should one add some extra constraints—or further information—on data structure $\mathcal{A}$ beyond those contained in the specification SP?

Typically, the answer to the latter question is affirmative: first-order logic is not strong enough to exclude all nonintended models of a specification, and hence nonintended implementations of the specified module. Similar problems arise also with specifications intended to be loose though not "too loose."

Various authors introduce some special strong additional requirements to cope with this problem. For instance, the category-theoretic initial (also, terminal) alge-

*Received by the editors June 10, 1997; accepted for publication (in revised form) August 23, 2000; published electronically May 16, 2001.
http://www.siam.org/journals/sicomp/30-6/32252.html
†LITA, Université de Pau, France, and Institute of Informatics, Białystok University of Technology, Białystok, Poland (mirkowska@pjwstk.waw.pl, salwicki@mimuw.edu.pl).
‡Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland (marians@ipipan.waw.pl). The work of the third author was partially supported by Poland's Scientific Research Committee under grant 2 P301 007 04.
§Institute of Informatics, Warsaw University, and Institute of Computer Science PAS, Warsaw, Poland (tarlecki@mimuw.edu.pl).

bra semantics approach has attracted a lot of attention. In these approaches, one selects as the semantics of a specification only initial (respectively, terminal) objects in the category of all algebras that satisfy given specification axioms; see, e.g., [4], [2], [18]. Another option is to require the algebras to be reachable (generated by the empty set, or standard—see the "generation principle" of [1]). Reachable algebras cannot be characterized by any set of first-order axioms, since every reachable algebra has an elementary extension that is not reachable, by direct application of the Skolem–Löwenheim theorem. None of these extra requirements (initiality, terminality, reachability) is expressible in first-order logic or its fragments (e.g., equational logic) often used in algebraic specifications. They can, however, be expressed in some stronger logics, for instance, in various versions of algorithmic logic (see [12]) or infinitary logics with proof systems involving infinitary proof rules (e.g., a version of the ω-rule) or in the second-order logic with the semantics where second-order variables range either over all subsets of the universe or only over finite subsets (weak second-order logic). Nevertheless, the classical first-order logic is of special interest here, at least because of its familiarity and since most of the contemporary mathematics has been done in it.

There are two possible ways to proceed: either we can restrict the class of models considered to some special structures only, characterized by some extralogical means, or we can work with a stronger logic. In this paper we promote and advocate the first possibility. However, we propose a "new" natural requirement of programmability (computability) of the models as the extralogical selection criterion. This seems more natural and friendly to programmers and to programming environment than other approaches proposed so far. By the Church thesis all possible implementations of any specification are programmable. Therefore, selecting programmable algebras as the only models considered is kind of a "minimal" requirement. It is perhaps surprising that this natural possibility has not been properly exploited in the theory and practice of algebraic specifications. The surprise is even more evident in light of a whole line of interesting results by Bergstra and Tucker [2] on the interplay between computability and equational specifications and on the expressive power of the initial algebra specifications.

To reiterate, we propose the classical first-order logic with the programmable (recursive) algebra semantics to be used for algebraic specifications.

It should be emphasized here with no need of any further details that the programmability requirement is natural and comprehensible to potential software developers, specifiers, and validators, as perhaps opposed to the abstract category-theoretic initiality or terminality criteria. A serious demand for detailed study of this kind of requirements follows, for instance, from a work of Sannella and Tarlecki [14] on a "gap" between the usual viewpoint of algebraic specification based on arbitrary models and that of programming frameworks (such as Extended ML [8]) more realistically based on programmable models. Traditionally in the area, abstract many-sorted algebras are used to model programs: the representation of data is arbitrary and operations are modelled as functions. Formal development of software systems from specifications calls for restriction of these general concepts to the kind of models that underlie semantics of programming languages. For example, the semantics of Standard ML [10] uses rather concrete models, where data values are represented as closed constructor terms and operations are represented as "closures."

We believe that the programmable reachable algebras provide an interesting general framework for algebraic specifications. In this note we present some technical

results to justify this belief and to start a proper investigation of such a framework.

2. Terminology and notation. In this paper we consider specifications as classical first-order (or *elementary*) theories. Unlike in the classical model-theory, we work with many-sorted algebras; a many-sorted algebra consists of a carrier set split into a family of nonempty carriers named by sorts with (a finite number of) operations on them. The carriers and operations of an algebra are named in the algebra's signature. Zero-ary operations are just the distinguished elements, interpreting constants of the language according to the algebra's signature. In general, the operations may be partial functions on their sorts. Throughout the paper we work with first-order logic with equality, i.e., formulas include equalities between terms of the same sort (for each sort) interpreted as the identity of the denotations of the terms. Occasionally we simplify the exposition by leaving the reader with many details of rigorous "sorting" of the formal expressions whose content is sketched only informally.

DEFINITION 2.1. *An algebra (in general, a relational structure) is called* programmable *(or* recursive*, or* computable*) whenever all of its carriers are computable sets and all of its operations (in the general case, relations) are computable functions defined on computable domains.*

We will be talking only about countable algebras. Therefore we can always consider beforehand an isomorphic copy of a given algebra built on the natural numbers and then define the algebra to be recursive whenever this copy is recursive. For example, the natural numbers with the usual arithmetic operations form a programmable algebra $(N; 0, suc, +, \times)$. Given a nonrecursive function f, the algebra $(N; 0, suc, +, \times, f)$ is not programmable.

We illustrate the ideas and results of this paper with various data types of stacks and their first-order specifications. We focus on the stacks because of their relevance for implementing recursive computations and modelling recursively defined collections of data, and because they provide a convenient example that we hope should be familiar to the reader. Stack theory can be formalized in various forms, essentially different in many ways from the metamathematical viewpoint. All of them share in their signature two sorts (one for elements and another for stacks) and the usual stack operations.

Given a finite or infinite set E, by *the true stack theory on* E we mean the first-order theory of the model with two-sorted carrier set E, E^*, where E^* is the set of all finite sequences of the members of E, and the usual operations *top*, *pop*, *push*, the distinguished *empty* sequence, and with all the elements of E as constants in the signature. In this paper we typically consider at most countably infinite E. The operations *top* and *pop* are undefined on *empty*. Here are the usual definitions for each $e \in E$ and $s = (e_0, \ldots, e_n) \in E^*$:

$$\begin{aligned} push(e, s) &= (e_0, \ldots, e_n, e), \\ top(s) &= e_n \quad \text{for nonempty } s, \\ pop(s) &= (e_0, \ldots, e_{n-1}) \quad \text{for nonempty } s. \end{aligned}$$

This yields a family of different true stack theories, one for each E. Modulo the names of the constants used, we in fact consider here one such theory for each (finite or infinite) cardinality of E. Each of them is formally a theory over a different signature (following the usual terminology in mathematical logic, we will say a *language* of a theory, rather than its signature).

In the countably infinite case, one can think of E as of the set of natural numbers. However, the members of E need not be the natural numbers, since the signature of the model does not contain any arithmetic operations.

By *the axiomatic theory of stacks on* E we mean the (classical first-order) consequences of the universal closure of the following axioms in the (first-order) language corresponding to the above model:

$$\begin{array}{ll} \text{(Ax1)} & push(x,s) \neq empty, \\ \text{(Ax2)} & top(push(x,s)) = x, \\ \text{(Ax3)} & pop(push(x,s)) = s, \\ \text{(Ax4)} & s \neq empty \rightarrow push(top(s), pop(s)) = s. \end{array}$$

This theory is a proper subtheory of the true stack theory on E. In particular, the true stack theory, but not the axiomatic stack theory, admits the following *scheme of structural induction* for every first-order formula φ:

$$[\varphi(empty) \wedge \forall x \, \forall s \, (\varphi(s) \rightarrow \varphi(push(x,s)))] \rightarrow \forall s \, \varphi(s).$$

By *the true theory of stacks on the natural numbers* we mean the first-order theory of the model $(N, N^*; 0, suc, +, \times, top, pop, push, empty)$ with the usual meaning of the arithmetic operations on the set N of natural numbers. The Peano axioms for the sort of elements and axioms (Ax1–Ax4) for stacks form yet another formalization, which we will call the *axiomatic stack theory on the natural numbers*.

Let us point out that here we mean the Peano axioms with the scheme of elementary induction

$$[\varphi(0) \wedge \forall x \, (\varphi(x) \rightarrow \varphi(x+1))] \rightarrow \forall x \, \varphi(x)$$

for every first-order formula φ. By Peano Arithmetic we mean the (classical) first-order consequences of the usual axioms for $0, suc, +, \times$ with the scheme of elementary induction.

3. Reachable algebras.

DEFINITION 3.1. *An algebra is called* reachable *if it is generated from it's underlying constants by finitely many successive applications of the algebra's operations.*

An algebra $\mathcal{A}$ is generated by a subset X of the carrier of $\mathcal{A}$ (a set of generators) if all the elements of $\mathcal{A}$ are the values of Σ-terms under some valuation of variables into X. Σ is the signature of $\mathcal{A}$. Now $\mathcal{A}$ is reachable iff it is generated by the empty set of generators.

Equivalently, an algebra is reachable if each element of its universe (carrier sets) is the denotation of a closed (ground) term in the (first-order) signature of the algebra (cf. the notion of a Herbrand model in model-theory). The reachable algebra of natural numbers, the only reachable model of Peano Arithmetic, is usually called the *standard model*. One says the reachable algebras have "no junk" property: they contain all data you can talk about in the algebra's formal language and nothing more. This does not imply, of course, another useful property of "no confusion" (or "unambiguity," or "the unique name") that different terms denote distinct data items, unless those terms are forced to be identified by the specification axioms.

Of course, reachable algebras need not be programmable in general. For instance, the nonprogrammable algebra $(N, 0, suc, f)$ with nonrecursive f mentioned above is reachable. In this paper we show that for a sufficiently rich theory its reachable

models can be distinguished in the class of programmable models by a single first-order axiom. This is based on a deep result of Tennenbaum [17] that *every recursive model of Peano Arithmetic (with the first-order induction scheme) is standard.*

Roughly, a first-order theory T is sufficiently rich in this context if Peano Arithmetic can be recursively interpreted in T and T has the finite sequence coding property. Let us recall the necessary notions of mathematical logic.

For the concept of an interpretation we refer, for instance, to Shoenfield [15]. We say that Peano Arithmetic can be interpreted in T whenever there are formulas $I_N(\cdot)$, $I_0(\cdot)$, $I_S(\cdot,\cdot)$, $I_{Add}(\cdot,\cdot,\cdot)$, and $I_{Mult}(\cdot,\cdot,\cdot)$ in the language of T such that T proves all the Peano axioms relativized to the interpretation I. An interpretation defines a model of Peano Arithmetic within every model of T: I_N defines its carrier, while the other formulas define the operations (or their graphs) of arithmetic, respectively. An interpretation is recursive if it defines a recursive model of Peano Arithmetic within every recursive model of T.

Given a theory with an interpretation of (enough of) arithmetic, one can think of sequences indexed by the natural numbers of the interpretation, i.e., by the elements of the carrier of the interpretation. Then one can talk about possible coding of initial segments of such sequences by single elements. In the following definition, intuitively, *BETA* is a uniform projection (decoding) function: for a sequence $a_0, a_1, \ldots, a_l$, there is a code u such that $BETA(u, j) = a_j$ for each $j \leq l$. Let us stress once more though that the "length" of the sequence and the "indexing subscripts" are numbers only in the sense of the interpretation of arithmetic.

DEFINITION 3.2. *We say that a theory T with an interpretation of arithmetic has* the finite sequence coding property *if there is a binary function BETA definable in T such that*

for every formula $A(x, y, parameters)$ in the language of T, if T proves
"for each x of the arithmetic interpretation universe I_N, there exists exactly one y such that $A(x, y, parameters)$,"
then T also proves the following:
"for each l of I_N, there exists a u such that for each $j \leq l$, $A(j, BETA(u, j), parameters)$."

Models of a theory with the finite sequence coding property should be compared with the concept of arithmetical universes of Harel [6]. Harel's arithmetical universes are equipped with a copy of the standard model of arithmetic, whereas we consider algebras equipped with a copy of just a model of Peano Arithmetic. The latter turns out to be standard under the assumptions of the following theorem.

THEOREM 3.3. *Let T be a first-order theory over a finite language such that*

1. *Peano Arithmetic can be recursively interpreted in T and*
2. *T has the finite sequence coding property.*

Then there is a single sentence σ_{reach} in the language of T such that every recursive model of T satisfying σ_{reach} is reachable.

Proof. To prove the above theorem we take the following sentence as σ_{reach}:

$$\forall x\, \exists u\, (u \text{ codes a path of reaching } x \text{ from the generators}).$$

More precisely, though still informally, the formula under the two quantifiers says the following: u codes a sequence of values in a given model M such that x is the last element of the sequence and each element in the sequence is either a denotation of a constant in M or can be obtained from earlier values in the sequence by application of one of the operations of M. Then consider a recursive model M of T. M contains a

recursive model of Peano Arithmetic. The latter has to be standard by Tennenbaum's result [17]. Therefore, if σ_{reach} holds in M, then M is reachable. □

Since the above sentence σ_{reach} holds in the standard model of the theory of stacks on the natural numbers (which contains Peano Arithmetic and clearly has the sequence coding property) the above theorem gives us the following corollary.

COROLLARY 3.4. *Every recursive model of the true theory of stacks on natural numbers is reachable.*

It now follows easily that every programmable model of the true stack theory on natural numbers is isomorphic to $(N, N^*; 0, suc, +, \times; top, pop, push, empty)$. Can anyone supply a better, while so friendly, specification of this data type?

The message can now be put forward as follows. For any sufficiently rich specification, if one restricts its semantics to programmable models only, then in order to ensure reachability, which in many cases means uniqueness, one simply needs to throw *implicitly* or *explicitly* the appropriate σ_{reach} into the specification. Given a sufficiently rich specification, in order to verify that all its programmable models are reachable it is necessary and sufficient to check whether the appropriate σ_{reach} is derivable from this specification. We have just applied this result to the true stack theory on natural numbers.

4. Nonreachable programmable algebras. Theorem 3.3 indicates that computability of a model of a sufficiently rich specification implies its reachability. In this section we provide examples of specifications (theories) that are not rich enough and admit recursive nonreachable models.

As an easy warm-up example consider the usual specification axioms for successor:

$$succ(x) \neq 0,$$
$$succ(x) = succ(y) \rightarrow x = y.$$

Any linear order of type $\omega + (\omega^* + \omega)$ provides a model of these axioms in the obvious way (with no claim of induction). Clearly, this model is recursive but nonreachable.

There are several ways of constructing a recursive nonreachable model of the true stack theory (as well as of its axiomatic mutations) on a finite set of elements. The present research has evolved from a definition of such a model written as a program in [11].

Here we give details of one possible construction of a recursive nonreachable model of the axiomatic stack theory on a finite E.

Let us first recall the Gödel's recursive function β such that for any natural numbers $a_1, \ldots, a_n$, there exists an $a \in N$ with $\beta(a, i) = a_i$, for each $i = 1, \ldots, n$, and with $\beta(a, 0) = n$. Moreover, $\beta(a, i) \leq a \dot{-} 1$. One can introduce the following sequence coding function:

$$SC(a_1, \ldots, a_n) \quad \stackrel{df}{=} \quad \mu x\,(\beta(x, 0) = n \wedge \beta(x, 1) = a_1 \wedge \cdots \wedge \beta(x, n) = a_n).$$

Define also the length function $lh(a) = \beta(a, 0)$, and the projection functions $(a)_i = \beta(a, i)$, for $i = 1, \ldots, lh(a)$. For $n = 0$, $\beta(0, i) = 0$, and so for the empty sequence ε we have $SC(\varepsilon) = 0$. We also need

$$Seq(a) \quad \stackrel{df}{=} \quad a \textit{ is a code of a sequence of length } lh(a).$$

The reader is referred to [15] for further details.

Now we define the model. Let $\{0,1\}$ be its carrier set of the element sort. The reader can easily modify this construction to any finite set of elements. Let its stack carrier be the union $S \cup S_0 \cup S_1$ of the following three sets of 0–1 sequences (finite or infinite):

- S is the set of all the finite 0–1 sequences;
- S_0 is the set of all the infinite 0–1 sequences stabilizing on 0;
- S_1 is the set of all the infinite 0–1 sequences stabilizing on 1.

Their union is recursive since they can be viewed as copies of the following sets of finite 0–1 sequences preceded by one of the markers 0, 1, or 2 and by their lengths. We use 0 as a marker for S_0, 1 for S_1, and 2 for S. Look at S_0 as the set of all finite sequences with 1 at the end, followed by infinitely many 0's. Allow the empty sequence there too—although it has no 1 at the end, it is needed to cover the case of the infinite sequence consisting only of 0's. Such infinite sequences can be coded up by pairs $\langle 0, a\rangle$ with sequence numbers a. The length $lh(a)$ indicates the position of the last 1. $lh(a) = 0$ indicates that there is no 1 at all. Similarly, deal with S_1 by indicating the position of the last 0 or indicating there is no 0 at all. Formally:

$$S \cup S_0 \cup S_1 = \{\langle \delta, a\rangle \mid \delta = 0,1,2 \wedge Seq(a) \wedge \forall 1 \le i \le lh(a)\,((a)_i = 0 \vee (a)_i = 1) \\ \wedge\ (lh(a) > 0 \to (a)_{lh(a)} \neq \delta)\}.$$

Define $empty = \langle 2, SC(\varepsilon)\rangle$, and for each $\delta = 0,1,2$ and each sequence code a of $(a_1, a_2, \ldots, a_{lh(a)})$ define the operations

- $top(\langle \delta, a\rangle) = \begin{cases} a_1 & \text{if } lh(a) > 0, \\ 0 & \text{if } lh(a) = 0 \text{ and } \delta = 0, \\ 1 & \text{if } lh(a) = 0 \text{ and } \delta = 1, \\ \text{undefined} & \text{otherwise.} \end{cases}$

- $pop(\langle \delta, a\rangle) = \begin{cases} \langle \delta, SC(a_2, \ldots, a_{lh(a)})\rangle & \text{if } lh(a) > 0, \\ \langle \delta, a\rangle & \text{if } lh(a) = 0 \text{ and } \delta \neq 2, \\ \text{undefined} & \text{otherwise.} \end{cases}$

- $push(e, \langle \delta, a\rangle) = \begin{cases} \langle \delta, SC(e, a_1, \ldots, a_{lh(a)})\rangle & \text{if } lh(a) > 0 \text{ or } \delta \neq e, \\ \langle \delta, a\rangle & \text{if } lh(a) = 0 \text{ and } \delta = e. \end{cases}$

LEMMA 4.1. *$(\{0,1\}, S \cup S_0 \cup S_1; top, pop, push, empty, 0, 1)$ is a recursive nonreachable model of the axiomatic stack theory.*

Proof. The proof has an easy verification. □

5. Nonreachable programmable models of decidable theories. In this section we show that there exists a nonreachable recursive model of the true stack theory on any given finite set E. We get it as an application of a general method (Corollary 5.4). The argument below relies heavily on the following theorem (known in the folklore of the field).

THEOREM 5.1 (the computable model existence theorem). *Every consistent decidable theory has a recursive model.*

Proof. One can "effectivize" Henkin's proof of the completeness theorem for the classical first-order logic. Some details are given in [7] and related results in [9]—for the reader's convenience we sketch a full proof in the appendix. The key step is an effective version of the celebrated Lindenbaum–Tarski theorem: every decidable consistent theory has a decidable complete extension in the same language. □

DEFINITION 5.2. *A theory is* reachably unambiguous *if all its reachable models are elementarily equivalent.*

It is easy to see that many interesting and useful theories are reachably unambiguous, e.g., Peano Arithmetic and Presburger Arithmetic. This is also the case for the stack theory in any of the above formalizations. In all these cases the theory has just one reachable model up to an isomorphism.

COROLLARY 5.3. *If T is a reachably unambiguous, incomplete, and decidable theory, then T has a recursive nonreachable model.*

Proof. In the algorithm of completion of T following Tarski [16], one can keep control on which of either σ or its negation $\neg\sigma$ gets into the completion, for a sentence σ such that neither $T \vdash \sigma$ nor $T \vdash \neg\sigma$. Since T is incomplete and reachably unambiguous, there is a sentence σ_0 such that neither $T \vdash \sigma_0$ nor $T \vdash \neg\sigma_0$, although σ_0 holds in every reachable model of T. Thus, a recursive model of the completion of T with $\neg\sigma_0$, built as in the proof of Theorem 5.1 above, cannot be reachable. □

COROLLARY 5.4. *Suppose T is a first-order theory. Let $c_0, c_1, c_2, \ldots$ be all the closed terms of the language of T, and let c be a new individual constant. If $T \cup \{\, c \neq c_n \mid n \text{ natural}\}$ is a consistent decidable theory, then T has a recursive nonreachable model.*

Proof. The proof is obvious by Theorem 5.1. □

We establish some results concerning decidability of the true stack theory in order to apply Corollary 5.4.

THEOREM 5.5. *Given a finite set E, the true stack theory on E is decidable.*

Proof. We show an interpretation of the true stack theory on E in Presburger Arithmetic (the first-order theory of $(N, suc, +, 0)$) which is well known to be decidable; see [13]. In fact, we show a somewhat stronger result by defining in $(N, suc, +, 0)$ an isomorphic copy of the model $(E, E^*; top, pop, push, empty, \{e\}_{e\in E})$.

Let k be the cardinality of E. We define the two-sorted carrier of our interpretation and its operations as follows with the usual notation for operations and relations definable in Presburger Arithmetic. ($\underline{k}$ denotes the kth numeral, i.e., $\underline{k}$ abbreviates $\underbrace{suc(\ldots suc(}_{k \text{ times}}\underline{0})\ldots)$).

- $U_E^I(e) \stackrel{df}{\equiv} e = \underline{0} \vee e = suc(\underline{0}) \vee \cdots \vee e = \underline{k-1}$;
- $U_S^I(s) \stackrel{df}{\equiv} s \geq \underline{k}$;
- $empty^I \stackrel{df}{=} \underline{k}$;
- $push^I(e, s) \stackrel{df}{=} \underline{k} \times (s - \underline{k}) + e + \underline{k} + 1$;
- $top^I(s) \stackrel{df}{=} (s - \underline{k} - 1) \bmod \underline{k}$, for $s > \underline{k}$;
- $pop^I(s) \stackrel{df}{=} [(s - \underline{k} - 1) \operatorname{div} \underline{k}] + \underline{k}$, for $s > \underline{k}$.

The idea behind this construction is that the stacks are coded up by numbers in their k-adic expansions, shifted up by adding $k + 1$ to make disjoint room for the elements and for the empty stack. The top^I operation returns the last digit of the expansion which is just the remainder of dividing by k. Similarly, pop^I cuts off the last digit. This is legal here since division by a given numeral is expressible in Presburger Arithmetic. Consequently, the decidability procedure for Presburger Arithmetic does the job for the true stack theory on E. □

The above proof gives immediately the following complexity results inherited from Presburger Arithmetic. The double exponential upper bound is due to Ferrante and Rackoff [3]. There exists a decision procedure and a constant c such that it takes

at most $2^{2^{cn}}$ deterministic single-tape Turing machine space to decide a sentence of length n. This gives deterministic time upper bound $2^{2^{2^{dn}}}$; see also Fischer and Rabin [5]. Precise complexity bounds for the true theory of stacks, which we have established recently beyond the scope of the present paper, will appear elsewhere.

LEMMA 5.6. *Let TST denote the true stack theory on a given finite set E. Then the theory $TST \cup \{c \neq c_n \mid n \text{ natural}\}$ is decidable, where $c_0, c_1, c_2, \ldots$ are all the closed terms of the stack sort of the language of TST, and c is a new individual constant of the stack sort.*

Proof. We reduce the decidability problem for this theory to the decidability of Presburger Arithmetic (denoted by PrA below). Let T denote the extended theory $TST \cup \{c \neq c_n \mid n \text{ natural}\}$. By a reduction we mean an effective mapping f associating to each sentence σ of the language $L(T)$ a sentence $f(\sigma)$ of the language $L(PrA)$ such that $T \vdash \sigma$ iff $PrA \vdash f(\sigma)$. Thus a decision procedure whether σ is a theorem of T will be reduced to whether $f(\sigma)$ is a theorem of Presburger Arithmetic. This will imply the decidability of T since Presburger Arithmetic is decidable.

For each σ of $L(T)$, we define $f(\sigma) = \exists y \forall x [x > y \longrightarrow I\sigma(x/c)]$, where $I\sigma$ is the interpretation of σ in Presburger Arithmetic as introduced in the proof of Theorem 5.5, separately for each E, and $I\sigma(x/c)$ is the substitution of x for c in $I\sigma$. We show that f is a reduction, i.e., $T \vdash \sigma$ iff $PrA \vdash f(\sigma)$.

($\Longrightarrow$) Suppose $T \vdash \sigma$. At most finitely many axioms of the form $c \neq c_n$ may appear in a proof of σ in T. Let l be the greatest number (subscript) of a constant c_n appearing in this proof. Then $TST \cup \{c \neq c_0, c \neq c_1, c \neq c_2, \ldots, c \neq c_l\} \vdash \sigma$. Thus

$$TST \vdash [c \neq c_0 \wedge c \neq c_1 \wedge c \neq c_2 \wedge \cdots \wedge c \neq c_l \rightarrow \sigma].$$

Since c is a new constant, there are no axioms concerning c in TST. Therefore

$$TST \vdash \forall x\, [x \neq c_0 \wedge x \neq c_1 \wedge x \neq c_2 \wedge \cdots \wedge x \neq c_l \rightarrow \sigma(x/c)],$$

and so

$$PrA \vdash \forall x\, [x \neq Ic_0 \wedge x \neq Ic_1 \wedge x \neq Ic_2 \wedge \cdots \wedge x \neq Ic_l \rightarrow I\sigma(x/c)].$$

In the standard model of Presburger Arithmetic the latter implies $\forall x[x > c_m \longrightarrow I\sigma(x/c)]$, where c_m is the greatest of Ic_0, Ic_1, Ic_2,..., Ic_l. Here by Ic_i we denote the interpretation of the constant term c_i. Hence $\exists y \forall x\, [x > y \rightarrow I\sigma(x/c)]$ holds in the standard model of Presburger Arithmetic. By completeness of Presburger Arithmetic we get $PrA \vdash \exists y \forall x\, [x > y \rightarrow I\sigma(x/c)]$.

($\Longleftarrow$) Suppose $PrA \vdash \exists y \forall x\, [x > y \rightarrow I\sigma(x/c)]$. Let m be such a number that $\forall x\, [x > m \rightarrow I\sigma(x/c)]$ holds in the standard model of PrA. Hence,

$$PrA \vdash \forall x\, [x \geq \underline{k} \wedge x \neq Ic_0 \wedge x \neq Ic_1 \wedge x \neq Ic_2 \wedge \cdots \wedge x \neq Ic_l \rightarrow I\sigma(x/c)],$$

where k is the cardinality of E and l is large enough for all the numerals from $\underline{k}$ to $\underline{m}$ to occur among Ic_0, Ic_1, Ic_2,..., Ic_l. Therefore

$$PrA \vdash [c \geq \underline{k} \wedge c \neq Ic_0 \wedge c \neq Ic_1 \wedge c \neq Ic_2 \wedge \cdots \wedge c \neq Ic_l \rightarrow I\sigma(c)],$$

where c is a new constant. Hence

$$TST \ \cup \ \{c \neq c_0 \wedge c \neq c_1 \wedge c \neq c_2 \wedge \cdots \wedge c \neq c_l\} \vdash \sigma(c)$$

for a new constant c of the stack sort. We get $T \vdash \sigma(c)$, since T contains

$$TST \cup \{c \neq c_0 \wedge c \neq c_1 \wedge c \neq c_2 \wedge \cdots \wedge c \neq c_l\}. \qquad \square$$

COROLLARY 5.7. *The true stack theory on a given finite set E has a recursive nonreachable model.*

Proof. The proof follows from Lemma 5.6 and Corollary 5.4. $\square$

COROLLARY 5.8. *The axiomatic stack theory on any finite E with the (first-order) scheme of structural induction has a recursive nonreachable model.*

Proof. It follows immediately from the above, since this theory is contained in the true stack theory on E. $\square$

REMARK 5.9. *Presburger Arithmetic has a recursive nonreachable model.*

Proof. It follows from Corollary 5.4 by a similar argument as in the proof of Lemma 5.6. This has been known to the experts although never published. $\square$

6. Conclusion. In general a first-order specification (axiomatization) has too many models. There have been several approaches in order to restrict the semantics of specifications to intended models only. The initial algebra approach proposed by [4] in the mid 1970s has particularly attracted a lot of attention. In this paper we propose and advocate the programmable algebra approach, i.e., to consider the programmable algebras as the only models. That is, by a specification we mean a pair (Σ, T), where Σ is a signature and T is a list of axioms in the classical first-order logic (with equality) of signature Σ, and the models of such a specification under the proposed semantics are all computable Σ-algebras that satisfy the axioms T. Furthermore,

- we give a criterion assuring that any specification which satisfies the criterion has only reachable models. Often this means just one model up to isomorphism. This is the case of the axiomatic theory of stacks on natural numbers.
- We give some examples of specifications that do not satisfy the criterion and admit programmable models which are not reachable. This is the case of both axiomatic and true stack theory on any given finite set E.
- We give a rather powerful method of constructing programmable nonreachable models.

These results can be easily generalized to provide a criterion that ensures that computable models of a theory are generated by a set of generators whenever this set is definable by a first-order formula. For instance, we may always indicate a subset of sorts and consider the carriers of these sorts as such a definable set of generators. This slightly more general version would be needed, for instance, to deal with stack theories on infinite sets of elements, which then can be considered over a finite signature, without constants for elements.

We also show decidability of the true stack theory on a finite set of elements, a result of interest on its own. One can use it, for instance, to conclude that Peano Arithmetic is not interpretable in the theory of stacks on a finite set of elements.

We do not know if the axiomatic stack theory is decidable. It remains as an intriguing open question. We conjecture that given a set E, the true stack theory on E is contained in the axiomatic stack theory on E with the scheme of structural induction. (Since the other inclusion is obvious, this would mean that the true and axiomatic theories are equivalent in this case.)

7. Appendix. In this appendix we provide a proof of Theorem 5.1: every decidable consistent theory has a recursive (computable) model. In view of this paper, it

seems to be of considerable revitalized interest for use in the area of algebraic specification of software systems, their semantics, validation, and systematic development. It provides a source of supply of programmable algebras to various purposes. The result has been in the folklore for many years. Some variant of it was stated in [9] without proof and another one (a bit stronger than our formulation) in [7] with a sketch of the proof idea.

As throughout this paper, we assume the reader is familiar with the usual formalization of classical first-order logic and its semantics; see, for example, [15]. All the theories considered will be formalized in classical first-order logic with countably many symbols. For simplicity we give the result and our proof here for the single-sorted case.

Recall that a theory T is called *decidable* if T is a recursive set (of Gödel numbers) of sentences, i.e., if there is an effective procedure deciding in a finite number of steps whether a given sentence is a theorem of T or not. Let us recall also that a theory T is *complete* if for every sentence σ exactly one of the following holds: either $\sigma \in T$ or $\neg\sigma \in T$. For two theories T and T', we say that T' is an extension of T if T' contains T. By $L(T)$ we denote the language of T.

To prove the theorem we follow Henkin's proof of the completeness theorem for classical first-order logic with appropriate modifications. Let T be a consistent decidable theory. We extend T to another theory T' such that T' is decidable, complete, and Skolemized. The latter means it has a term witness for each existential statement derivable in this theory. The *canonical structure* will be constructed on the terms of the language of T'. Its interpretations of relation and function symbols will be defined according to what T' knows about them. We then prove the canonical structure is a model of T'. Thus it is a model of T too, since T is a subtheory of T'. The model turns out recursive because T' is a decidable theory. That is, the construction of the model is described recursively by the story of T'. To this end we need the following two lemmas. The first of them can be thought of as an effectivized Lindenbaum theorem.

LEMMA 7.1. *Every decidable consistent theory has a complete decidable and consistent extension in the same language.*

Sketch of proof (Lindenbaum–Tarski). Let T be a decidable consistent theory. Let $\varphi_0, \varphi_1, \varphi_2, \ldots$ be a recursive enumeration of all sentences of the language of T. Let T_0 be T itself. Given decidable theory T_n, define k_n as the least k such that $\neg\varphi_k$ is not a theorem of T_n. Then let T_{n+1} be the theory that results from adding φ_{k_n} to T_n as a new axiom. T_{n+1} is decidable, since T_{n+1} proves σ iff T_n proves $\varphi_{k_n} \to \sigma$. Now let T' be the theory which has as its axioms all of the axioms of these theories T_n, $n \geq 0$. One can show that T' is consistent, complete, and decidable. □

LEMMA 7.2 (Skolemization). *If T is a consistent decidable theory, then there is a consistent decidable extension T' of T with the language $L(T')$ containing $L(T)$ and such that if a sentence $\exists x\varphi(x)$ is a theorem of T, then there is a term t in $L(T')$ with $\varphi(t)$ belonging to T'.*

Proof. Let T be a consistent decidable theory. We construct by recursion a sequence of languages L_0, L_1, L_2, ... and a sequence of theories T_0, T_1, T_2, ... At each step we extend the language with a new constant and adjoin one new axiom to the theory. First, let us enumerate recursively all the existential sentences of $L(T)$: $\exists x\varphi_n(x)$, n natural. We take off with $L_0 = L(T)$ and $T_0 = T$. Now suppose we are given a consistent decidable theory T_n in $L(T_n)$. If $\exists x\varphi_n(x)$ is derivable in T_n, then we take L_{n+1} to be L_n plus a new constant c_n, and T_{n+1} to result from T_n by adding $\varphi_n(c_n)$ as an axiom. Otherwise, we take $L_{n+1} = L_n$ and $T_{n+1} = T_n$. Clearly, new

theory T_{n+1} is consistent.

To show that in the nontrivial case T_{n+1} is decidable we reduce the decidability problem for this theory to decidability of T_n. As in the proof of Lemma 5.6, here by a reduction we mean an effective mapping f associating with each sentence σ of the language $L(T_{n+1})$ a sentence $f(\sigma)$ of the language $L(T_n)$ such that $T_{n+1} \vdash \sigma$ iff $T_n \vdash f(\sigma)$. Thus a decision procedure whether σ is a theorem of T_{n+1} will be reduced to that of whether $f(\sigma)$ is a theorem of T_n. This will mean decidability of T_{n+1}, since T_n is decidable, by the inductive hypothesis. We take $f(\sigma) \stackrel{df}{=} \exists x[\varphi_n(x)\&\sigma(x/c_n)]$. Given a sentence σ in $L(T_{n+1})$ and a proof d of σ in T_{n+1}, we notice that the axiom $\varphi_n(c_n)$ can occur at most finitely many times in d. We can obtain a proof d' of $\exists x[\varphi_n(x)\&\sigma(x/c_n)]$ in T_n out of d as follows. Replace each occurrence of the axiom $\varphi_n(c_n)$ in d by $\exists x\varphi_n(x)$. Then introduce the prefix $\exists x[\varphi_n(x)\&\ldots]$ to each member (step) of d containing c_n, possibly renaming the other variables. Finally, substitute each occurrence of c_n with x. The other way around is proved similarly. Thus f is a reduction.

To complete the proof, let T' be the union of all the theories T_n, $n \geq 0$. Clearly, T' is consistent. It is decidable as a recursive union of recursive sets. □

Proof of Theorem 5.1. Let T be a consistent decidable theory. As the first step we construct a consistent, decidable, complete, and Skolemized extension T^* of T. To this end we construct recursively a sequence L_0, L_1, L_2, ... of languages and a sequence T_0, T_0', T_1, T_1', T_2, T_2', ... of theories as follows. We start off with $L_0 = L(T)$ and $T_0 = T$ and set T_0' to be an extension of T provided by Lemma 7.1. Suppose we are given a consistent and decidable theory T_n in language $L_n = L(T_n)$. We take $L_{n+1} = L(T')$ and $T_{n+1} = T'$ where T' is an extension of T_n provided by Lemma 7.2. Clearly, T_{n+1} is consistent and decidable. Now we take the union of all of them:

$$L^* = \bigcup_{n \geq 0} L_n$$

and

$$T^* = \bigcup_{n \geq 0} T_n = \bigcup_{n \geq 0} T_n'.$$

T^* is decidable, consistent, complete, and Skolemized.

As for the rest of the proof we only use a standard construction of the canonical model for T^* (see, e.g., [15], pp. 44–46). We represent the resulting equivalence relations on the natural numbers. Equivalence classes can be identified with their least (number) representatives. The resulting model is computable since T^* is decidable. □

Acknowledgments. The third author would like to thank Richard Kaye for a number of inspiring conversations. Thanks to an anonymous referee who suggested to improve the paper by including a proof of Theorem 5.1 as an appendix.

REFERENCES

[1] F. Bauer and H. Wössner, *Algorithmic Language and Program Development*, Springer-Verlag, Berlin, Heidelberg, New York, 1982.

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

[3] J. Ferrante and C. Rackoff, *A decision procedure for the first order theory of real addition with order*, SIAM J. Comput., 4 (1975), pp. 69–76.
[4] J.A. Goguen, J.W. Thather, and E. Wagner, *An initial algebra approach to the specification, correctness and implementation of abstract data types*, in Current Trends in Programming Methodology, R. Yeh, ed., Prentice-Hall, Englewood Cliffs, NJ, 1978, pp. 80–149.
[5] M.J. Fischer and M.O. Rabin, *Super-Exponential Complexity of Presburger Arithmetic*, SIAM-AMS Proc. 7, AMS, Providence, RI, 1974, pp. 27–41.
[6] D. Harel, *First Order Dynamic Logic*, Lecture Notes in Comput. Sci. 68, Springer-Verlag, Berlin, New York, 1979.
[7] V.S. Harizanov, *Pure computable model theory*, in Handbook of Recursive Mathematics, Yu. L. Ershov, S.S. Goncharov, A. Nerode, J.B. Remmel, and V.W. Marek, eds., Stud. Logic Found. Math. 138, North-Holland, Amsterdam, 1998.
[8] S. Kahrs, D. Sannella, and A. Tarlecki, *The definition of Extended ML: A gentle introduction*, Theoret. Comput. Sci., 173 (1997), pp. 445–484.
[9] T.S. Millar, *Foundations of recursive model theory*, Ann. Math. Logic, 13 (1978), pp. 45–72.
[10] R. Milner, R. Harper, and M. Tofte, *The Definition of Standard ML*, MIT Press, Cambridge, MA, 1991.
[11] G. Mirkowska and A. Salwicki, *The algebraic specifications do not have the Tennenbaum property*, Fund. Inform., 28 (1996), pp. 141–152.
[12] G. Mirkowska and A. Salwicki, *Algorithmic Logic*, PWN, Warsaw, Poland, and D. Reidel, Dordrecht, the Netherlands, 1987.
[13] M. Presburger, *Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen in welchem die Addition als einzige Operation hervortritt*, in Comptes Rendus du 1er Congrès des Mathematiciens des Pays Slaves, Warszawa, 1929, pp. 92–101.
[14] D. Sannella and A. Tarlecki, *Mind the Gap! Abstract Versus Concrete Models of Specifications*, Lecture Notes in Comput. Sci. 1113, Springer-Verlag, Berlin, 1996, pp. 114–134.
[15] J. Shoenfield, *Mathematical Logic*, Addison-Wesley, Reading, MA, 1967.
[16] A. Tarski, A. Mostowski, and R. Robinson, *Undecidable Theories*, North-Holland, Amsterdam, 1953.
[17] S. Tennenbaum, *Non-Archimedean models of arithmetic*, Notices Amer. Math. Soc., 1959, p. 270.
[18] M. Wirsing, *Algebraic specification*, in Handbook of Theoretical Computer Science, J. van Leeuwen, ed., Elsevier, Amsterdam, 1990, pp. 676–788.