ELSEVIER

Available online at www.sciencedirect.com

SCIENCE DIRECT®

The Journal of Logic and Algebraic Programming 62 (2005) 71–108

THE JOURNAL OF LOGIC AND ALGEBRAIC PROGRAMMING

www.elsevier.com/locate/jlap

Computable total functions on metric algebras, universal algebraic specifications and dynamical systems[☆]

J.V. Tucker[a], J.I. Zucker[b,*]

[a] *Department of Computer Science, University of Wales Swansea, Singleton Park, Swansea, Wales SA2 8PP, UK*
[b] *Department of Computing and Software, McMaster University, Hamilton, Ont., Canada L8S 4L7*

Received 14 November 2002; accepted 17 October 2003

Abstract

Data such as real and complex numbers, discrete and continuous time data streams, waveforms, scalar and vector fields, and many other functions, are fundamental for many kinds of computation. In the theory of data, such data types are modelled using *topological*, or *metric*, many-sorted algebras and continuous homomorphisms. A theory of such topological data types is needed to answer the general questions:

1. What are the computable functions on topological algebras?
2. What methods exist to axiomatically specify functions on topological algebras?
3. Can all computable functions be specified?

Such a theory seems to be in its infancy: there are many approaches to computability theory on general and specific spaces, and few approaches to specification theory. In some earlier papers, we have studied the questions 1 and 2 with the needs of data type theory in mind, and built a bridge between computations and specifications to try to answer 3. In this paper, we extend and combine several of our results, to prove new theorems that

(i) show the equivalence of some six deterministic or non-deterministic models of computation on various metric algebras and, in particular, on spaces $\mathbb{R}^n$ of real numbers;
(ii) provide finite *universal* algebraic specifications for all the functions that can be computably approximated on metric algebras and, in particular, on Euclidean n-space $\mathbb{R}^n$;
(iii) show the existence of finite universal algebraic specifications of computably approximable finite dimensional deterministic dynamical systems.

[☆] The research of the second author was supported by a grant from the Natural Sciences and Engineering Research Council (Canada).
[*] Corresponding author.
E-mail addresses: j.v.tckker@swansea.ac.uk (J.V. Tucker), zucker@mcmaster.ca (J.I. Zucker).

1567-8326/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.jlap.2003.10.001

A technical issue is the localisation of uniform continuity using exhaustions of open sets. We use specifications composed of conditional equations, inequalities and, for convenience, new exhaustion primitives, that define functions uniquely up to isomorphism.

Keywords: Many-sorted algebra; Topological algebra; Metric algebra; Uniform continuity; Computable function; Computable approximations; While programs; Countable choice; Conditional equations

1. Introduction

1.1. Overview

In the theory of data, abstract data types are modelled by many-sorted algebras and homomorphisms, and are specified axiomatically by equations and conditional equations. Most of the theory has been developed for data that can be *exactly* represented by *finite* information. These data types are discrete and countable. They are the data types for which exact digital computation is possible. There is a comprehensive theory of data that reveals intimate connections between computability, algebraic specification methods, and term rewriting. The power of the various specification methods has been shown to correspond with basic concepts from computability. See, for instance, the surveys by Meseguer and Goguen [27] and Stoltenberg-Hansen and Tucker [42], and the papers by Bergstra and Tucker [5–7], Meseguer et al. [29] and Khoussainov [22].

In contrast, consider the case of data types that do *not* possess exact finite representations and, therefore, require *infinite* representations. Examples are the real numbers $\mathbb{R}$, and various finite dimensional systems based on $\mathbb{R}^n$, including algebras of real number matrices; streams and waveforms; scalar and vector fields; and real or complex Banach or Hilbert spaces. Here, in contrast, the data types are continuous and uncountable. Such data types are fundamental in modelling physical dynamical systems and are the characteristic data types of analogue processing. What is the relationship between the computations on continuous data and the *exact* computations on discrete data that will ultimately *approximate* computations on continuous data? The theory is developing, but compared with the theory of discrete data types, little is known.

Now, such data types can be modelled by *topological* many-sorted algebras and continuous homomorphisms, or—to take a more restricted class, closer to the examples mentioned—by many-sorted algebras that are *metric spaces*. However, there are obstacles to progress with the theory of continuous data types. The rather basic questions:

1. What are the computable functions on topological algebras?
2. What methods exist to axiomatically specify functions on topological algebras?

need answers prior to the question:

3. Can all computable functions be specified?

Such a theory is in its infancy: there are *many* approaches to computability theory on general and specific spaces, and *few* approaches to specification theory. In [51,52,55] we have studied question 1. In [54] we have studied question 2 and built a bridge between computability and specifications, as a start to answer question 3. In this paper, we extend and combine these and other earlier results, to prove new theorems that

(i) show the equivalence of some 6 deterministic or non-deterministic models of computation on various metric algebras and, in particular, on spaces $\mathbb{R}^n$ of real numbers;
(ii) provide finite *universal* algebraic specifications for all the functions that can be computably approximated on metric algebras and, in particular, on spaces $\mathbb{R}^n$ of real numbers;
(iii) show the existence of finite universal algebraic specifications for computably approximable, finite dimensional, deterministic dynamical systems.

Here we extend our earlier results by removing conditions of global uniform continuity and compactness. Global uniform continuity simplifies considerably technical definitions of the computability of functions on spaces. In metric algebras, compactness implies that continuous functions are uniformly continuous. We consider the broader class of functions that are uniformly continuous in pieces, by "localising" the uniformities, necessary for computability, using *open exhaustions* (U, V), where V is a sequence of open subsets

$$V = (V_0, V_1, V_2, \ldots) \quad \text{and} \quad \bigcup_{p=0}^{\infty} V_p = U.$$

This leads to the notion of *effective local uniform continuity*. Exhaustions are an obvious and standard way of extending computability notions via localisation. In our case, the resulting theorems are much more useful, having applications to, say, partial functions on all of n-space $\mathbb{R}^n \to \mathbb{R}^m$, rather than on a compact n-cube $[a, b]^n \to \mathbb{R}^m$. This paper generalises the ideas and techniques of the earlier papers to exhaustions, and takes a wider view of the problems suggested by these results.

1.2. Abstract and concrete computability

There are many approaches to defining computability on topological and metric algebras. Theories may be divided into:

- *abstract computability theories*, in which computations are independent of data representations; and
- *concrete computability theories*, in which computations depend on chosen data representations.

Abstract computation theories are designed for all many-sorted algebras, and so enable us is to develop a number of special computability theories for algebras such as *rings and fields of real numbers* [2,13,50] and topological and metric algebras [51–53,55]. Many abstract models of computation have been defined and shown to be equivalent over general algebras; hence the theory of abstract computation is quite stable. For a comprehensive introduction to abstract computation, including a survey of its origins in the 1950s and principal literature, see our survey paper [52]. Here we will use mainly the "while"-array model of computation, the primary mathematical model of imperative programming, over these algebras.

Concrete computation theories are designed to analyse computability in terms of classical recursion theory on natural numbers via chosen representations of data and spaces. They have been used to study *computable analysis*, starting with Grzegorczyk's and Lacombe's study of computation on real numbers [14,15,24]. Some general approaches are due to Moschovakis [32], Pour-El and Richards [35], Weihrauch [57], Stoltenberg-Hansen and Tucker [41,42], Spreen [45,46] and Edelat [12]. The equivalence of most of these concrete approaches is proved for certain topological algebras in [43]. The study of concrete computability is, however, still not so well understood or stable. Nevertheless, on the real numbers the concrete models have all been shown to define the Grzegorczyk–Lacombe (GL) computable functions.

The connection between the abstract and concrete theories has been problematic. Recently, the situation has been clarified when, surprisingly, notions of limit processes, approximation, non-determinism and multivaluedness, have been shown to be necessary to bridge the gap between the two for general classes of algebras [3,4,51,55]. Here we use our "while"-array language with *non-deterministic countable choice*, i.e., a new assignment

$$\mathtt{x} := \mathsf{choose}\ \mathtt{z} : b(\mathtt{z}, \ldots)$$

of type nat, where z is a variable of type nat and b is a term of type bool.

In Section 3, we extend our general notion of *effective Weierstrass approximation* for total metric algebras [51] to a "localised" notion using exhaustions. We show that for connected exhaustions, on certain algebras, localised versions of "while" approximation, "while"-array approximation, and Weierstrass approximation are all equivalent (Theorem 3.2.19). In particular, for a certain *total metric algebra* $\mathscr{R}_t^N$ of real numbers, local Weierstrass approximation coincides with local $\mathbb{Q}$-polynomial approximation, and all these localised notions coincide further with GL-computability on $\mathbb{R}$ (Theorem 3.3.2).

In Section 4 we extend our main bridging theorem in [55] to the localised case. We show that for effectively locally uniformly continuous functions, and a wide class of metric algebras, approximation by "while"-array programs with countable choice is equivalent to a simple concrete Moschovakis-like "tracking" computational model (Theorem 4.2.13). All these results for computation on $\mathbb{R}$ are combined to obtain (Theorem 4.4.1):

Theorem. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a total function that is effectively locally uniformly continuous. Then the following are equivalent*:
(i) *f is GL-computable on $\mathbb{R}$*,
(ii) *f is effectively trackable on $\mathbb{R}$*,
(iii) *f is effectively locally $\mathbb{Q}$-polynomially approximable on $\mathbb{R}$*,
(iv) *f is effectively locally uniformly* while *approximable on $\mathscr{R}_t^N$*,
(v) *f is effectively locally uniformly* while-*array approximable on $\mathscr{R}_t^N$*,
(vi) *f is effectively uniformly* while-*array with countable choice approximable on $\mathscr{R}_p^N$*.

(The "locality" referred to here is with respect to a "standard exhaustion" of $\mathbb{R}$.) This gives us a stable foundation for the idea of a *locally computable function* based on exhaustions. Next we consider the specification of these functions.

1.3. Algebraic specifications

Algebraic specification methods characterise functions as the solutions of systems of algebraic formulae that are unique in some sense. By algebraic formulae, we mean *equations*

$$t(\mathrm{x}) = t'(\mathrm{x})$$

or *conditional equations*

$$t_1(\mathrm{x}) = t_1'(\mathrm{x}) \wedge \cdots \wedge t_k(\mathrm{x}) = t_k'(\mathrm{x}) \to t(\mathrm{x}) = t'(\mathrm{x}),$$

or, more generally, other formulae, based on these, and enjoying some algebraic properties or customised to the particular algebraic context. For example, in working in metric algebras, we have as standard the sort of real numbers to measure distance, so we will adapt the formulae to include

(i) *inequalities* between reals ($t_1 < t_2$),

and, using the exhaustions, we will adapt these further to include

(ii) *localisation* ($t_1 \in V_{t_2}$).

Taking (i) and (ii) together, we form specifications using *localised conditional equations and inequalities*, which provide unique solutions, as required.

In Section 5, we show (Theorem 5.1.3):

Theorem. *For each signature Σ and function type τ there exists a signature Σ^+ that extends Σ by functions only, and a finite set $E(\mathrm{z})$ of localised conditional equations and inequalities over Σ^+ with natural number variable* z *such that for any "*while*"-array procedure P over Σ, total metric Σ-algebra A, exhaustion (U, V) of A, and function f of type τ on A which is defined on U, if f is effectively locally approximable by P with respect to (U, V), then f is defined uniquely on A by $E(\bar{k})$, where $\bar{k}$ is a numeral effectively calculable from P. The specification $(\Sigma^+, E(\mathrm{z}))$ is computable from Σ and τ.*

We say that the specification $(\Sigma^+, E(\mathrm{z}))$ is a *universal specification* for the computably approximable functions of type τ over *all* metric Σ-algebras A. Of immediate interest is the case of the real numbers. Using the total metric algebra $\mathscr{R}_t^N$ with its corresponding signature $\Sigma = \Sigma(\mathscr{R}_t^N)$, and the results above, we can derive (Theorem 5.2.1):

Corollary. *There is a finite universal specification $(\Sigma^+, E(\mathrm{z}))$, consisting of conditional equations and inequalities over Σ^+, that uniquely defines all locally GL-computable total functions on $\mathbb{R}$.*

From this it is easy to derive (Theorem 5.3.1):

Corollary. *There is a finite universal specification $(\Sigma^+, E(\mathrm{z}))$, consisting of conditional equations and inequalities over Σ^+, that uniquely defines all locally GL-computable dynamical systems on $\mathbb{R}^n$.*

Thus, a physical system that can be algorithmically approximated in finite dimensions can be defined by a finite system of algebraic formulae. Indeed, for each dimension n, there exists a universal finite system of algebraic formulae that defines all the n-dimensional systems. We discuss the implications of this last corolllary later (in Section 6.3).

Finally, in Section 6, we give some concluding remarks.

This paper is part of our series on abstract computability theory on many-sorted algebras and its applications, starting in [49] and most recently surveyed in [52]. Readers should be familiar with the theory of computing by while programs on abstract many-sorted algebras (as in [52]), and our papers [51,53,55] should preferably be at hand for background information and some of the proofs.

2. Topological partial algebras and continuity

We briefly survey the basic concepts of topological and metric many-sorted partial algebras. More details can be found in [51,52,55].

2.1. Basic algebraic definitions

A *signature* Σ (for a many-sorted partial algebra) is a pair consisting of (i) a finite set ***Sort***(Σ) of *sorts*, and (ii) a finite set ***Func***(Σ) of (*basic*) *function symbols*, each symbol F having a *type* $s_1 \times \cdots \times s_m \to s$, where $s_1, \ldots, s_m, s \in \textbf{\textit{Sort}}(\Sigma)$; in that case we write $F : s_1 \times \cdots \times s_m \to s$. (The case $m = 0$ corresponds to *constant symbols*.)

A *Σ-product type* has the form $u = s_1 \times \cdots \times s_m$ $(m \geqslant 0)$, where $s_1, \ldots, s_m$ are Σ-sorts.

A partial *Σ-algebra A* has, for each sort s of Σ, a non-empty *carrier set A_s* of sort s, and for each Σ-function symbol $F : u \to s$, a partial function $F^A : A^u \to A_s$, where, for the Σ-product type $u = s_1 \times \cdots \times s_m$, we write $A^u =_{df} A_{s_1} \times \cdots \times A_{s_m}$. (The notation $f : X \to Y$ refers in general to a partial function from X to Y.)

The algebra A is *total* if F^A is total for each Σ-function symbol F.

In this paper *the default assumption will be that "algebra" refers to partial algebra.*

Given an algebra A, we write $\Sigma(A)$ for its signature.

Example 2.1.1

(a) The algebra of *booleans* has the carrier $\mathbb{B} = \{\texttt{tt}, \texttt{ff}\}$ of sort bool. The signature $\Sigma(\mathscr{B})$ and algebra $\mathscr{B}$ can be displayed as follows:

signature	$\Sigma(\mathscr{B})$
sorts	bool
functions	true, false $:\to$ bool,
	and, or : bool$^2 \to$ bool,
	not : bool $\to$ bool
end	

and

algebra	$\mathscr{B}$
carriers	$\mathbb{B}$
functions	$\texttt{tt}, \texttt{ff} :\to \mathbb{B}$,
	and$^{\mathscr{B}}$, or$^{\mathscr{B}}$: $\mathbb{B}^2 \to \mathbb{B}$,
	not$^{\mathscr{B}}$: $\mathbb{B} \to \mathbb{B}$
end	

(b) The algebra $\mathscr{N}_0$ of naturals has a carrier $\mathbb{N}$ of sort nat, together with the zero constant and successor function:

algebra	$\mathscr{N}_0$
carriers	$\mathbb{N}$
functions	$0 :\to \mathbb{N}$,
	$\mathsf{S} : \mathbb{N} \to \mathbb{N}$
end	

(c) The ring $\mathscr{R}_0$ of reals has a carrier $\mathbb{R}$ of sort real:

algebra	$\mathscr{R}_0$
carriers	$\mathbb{R}$
functions	$0, 1 :\to \mathbb{R}$,
	$+, \times : \mathbb{R}^2 \to \mathbb{R}$,
	$- : \mathbb{R} \to \mathbb{R}$
end	

(d) The field $\mathscr{R}_1$ of reals is formed by adding the multiplicative inverse to the ring $\mathscr{R}_0$:

algebra	$\mathscr{R}_1$
import	$\mathscr{R}_0$
functions	inv$^{\mathscr{R}}$: $\mathbb{R} \to \mathbb{R}$
end	

where

$$\mathsf{inv}^{\mathscr{R}}(x) = \begin{cases} 1/x & \text{if } x \neq 0, \\ \uparrow & \text{otherwise.} \end{cases}$$

Example (d) is a partial algebra.

Throughout this work we make the following ***Instantiation Assumption*** about the signatures Σ:

For every sort s of Σ, there is a closed term of that sort, called the default term $\boldsymbol{\delta}^s$ of that sort.

2.2. Adding booleans: standard signatures and algebras

Definition 2.2.1 (*Standard signature*). A signature Σ is *standard* if (i) it contains the signature of booleans, i.e., $\Sigma(\mathscr{B}) \subseteq \Sigma$; and (ii) the function symbols of Σ include the *conditional* $\mathsf{if}_s : \mathsf{bool} \times s^2 \to s$ for all sorts s of Σ other than bool.

Given a standard signature Σ, a sort of Σ is called an *equality sort* if Σ includes an *equality operator* $\mathsf{eq}_s : s^2 \to \mathsf{bool}$.

Definition 2.2.2 (*Standard algebra*). Given a standard signature Σ, a Σ-algebra A is *standard* if (i) it is an expansion of $\mathscr{B}$; (ii) the conditional operator if_s on each sort s has its standard interpretation in A; and (iii) the equality operator eq_s is interpreted as a *partial identity* on each equality sort s, i.e., for any two elements of A_s, if they are identical, then the operator at these arguments returns $\mathtt{tt}$ if it returns anything; and if they are not identical, it returns $\mathtt{ff}$ if anything.

Two typical examples of partial identity as an interpretation of eq_s are: (1) total equality, where equality is assumed to be "decidable" at sort s; this is appropriate, for example, when $s = \mathsf{nat}$; (2) the case

$$\mathsf{eq}_s^A(x, y) = \begin{cases} \uparrow & \text{if } x = y, \\ \mathtt{ff} & \text{otherwise,} \end{cases}$$

where equality is "co-semidecidable"; this is appropriate, for example, in our partial real algebra $\mathscr{R}_p$ (Example 2.2.4(c) below).

Any many-sorted signature Σ can be *standardised* to a signature $\Sigma^{\mathscr{B}}$ by adjoining the sort bool together with the standard boolean operations; and, correspondingly, any algebra A can be standardised to an algebra $A^{\mathscr{B}}$ by adjoining the algebra $\mathscr{B}$ and the conditional operator at all sorts, and, where required, the (partial) equality operators at certain sorts.

Throughout this paper, we will assume:

the signature Σ, and the Σ-algebra A, are standard.

Example 2.2.3

(a) A *standard algebra of naturals $\mathscr{N}$* is formed by standardising the algebra $\mathscr{N}_0$ (Example 2.1.1(b)), with (total) equality and order operations on $\mathbb{N}$:

algebra	$\mathscr{N}$
import	$\mathscr{N}_0, \mathscr{B}$
functions	$\mathsf{if}_{\mathsf{nat}}^{\mathscr{N}} : \mathbb{B} \times \mathbb{N}^2 \to \mathbb{N}$,
	$\mathsf{eq}_{\mathsf{nat}}^{\mathscr{N}}, \mathsf{less}_{\mathsf{nat}}^{\mathscr{N}} : \mathbb{N}^2 \to \mathbb{B}$
end	

(b) The *standardised ring of reals* (cf. Example 2.1.1(c)):

algebra	$\mathscr{R}_0^B$
import	$\mathscr{R}_0, \mathscr{B}$
functions	$\mathsf{if}_{\mathsf{real}}^{\mathscr{R}} : \mathbb{B} \times \mathbb{R}^2 \to \mathbb{R}$,
end	

(c) *A standard partial algebra $\mathscr{R}_p$ on the reals* is formed similarly by standardising the field $\mathscr{R}_1$, itself a partial algebra (Example 2.1.1(d)), with partial equality and order operations on $\mathbb{R}$:

algebra	$\mathscr{R}_p$
import	$\mathscr{R}_1, \mathscr{B}$
functions	$\mathsf{if}_{\mathsf{real}}^{\mathscr{R}} : \mathbb{B} \times \mathbb{R}^2 \to \mathbb{R}$,
	$\mathsf{eq}_{\mathsf{real}}^{\mathscr{R}}, \mathsf{less}_{\mathsf{real}}^{\mathscr{R}} : \mathbb{R}^2 \to \mathbb{B}$
end	

where

$$\mathsf{eq}_{\mathsf{real}}^{\mathscr{R}}(x, y) = \begin{cases} \uparrow & \text{if } x = y, \\ \mathtt{ff} & \text{if } x \neq y, \end{cases}$$

and

$$\mathsf{less}_{\mathsf{real}}^{\mathscr{R}}(x, y) = \begin{cases} \mathtt{tt} & \text{if } x < y, \\ \mathtt{ff} & \text{if } x > y, \\ \uparrow & \text{if } x = y. \end{cases}$$

The significance of these partial equality and order operations, in connection with computability and continuity, is discussed in [55].

2.3. Adding counters: N-standard signatures and algebras

Definition 2.3.1 (*N-standard signature*). A signature Σ is N-*standard* if (i) it is standard, and (ii) it contains the standard signature of naturals (2.2.4(a)), i.e., $\Sigma(\mathscr{N}) \subseteq \Sigma$.

Definition 2.3.2 (*N-standard algebra*). Given an N-standard signature Σ, a corresponding Σ-algebra A is N-*standard* if it is an expansion of $\mathscr{N}$.

Any standard signature Σ can be N-*standardised* to a signature Σ^N by adjoining the sort nat and the operations 0, S, $\mathsf{eq}_{\mathsf{nat}}$, $\mathsf{less}_{\mathsf{nat}}$ and $\mathsf{if}_{\mathsf{nat}}$. Correspondingly, any standard Σ-algebra A can be N-*standardised* to an algebra A^N by adjoining the carrier $\mathbb{N}$ together with the corresponding standard functions.

Example 2.3.3. We can N-standardise the standard partial real algebra $\mathscr{R}_p$ (2.2.4(c)) to form the algebra $\mathscr{R}_p^N$.

2.4. Adding arrays: algebras A^* of signature Σ^*

Given a standard signature Σ, and standard Σ-algebra A, we expand Σ and A in two stages: (1) N-standardise these to form Σ^N and A^N, as in Section 2.3; and (2) define, for each sort s of Σ, the carrier A_s^* to be the set of *finite sequences* or *arrays* a^* over A_s, of "starred sort" s^*.

The resulting algebras A^* have signature Σ^*, which extends Σ^N by including, for each sort s of Σ, the new starred sorts s^*, and certain new function symbols to read and update arrays. Details are given in [51,52].

2.5. Topological partial algebras

We now add topologies to our partial algebras, with the requirement of continuity for the basic partial functions.

Definition 2.5.1. Given two topological spaces X and Y, a partial function $f : X \to Y$ is *continuous* if for every open $V \subseteq Y$, the pre-image

$$f^{-1}[V] =_{df} \{x \in X \mid x \in \boldsymbol{dom}(f) \text{ and } f(x) \in V\}$$

is open in X.

Definition 2.5.2
(a) A *topological partial Σ-algebra* is a partial Σ-algebra with topologies on the carriers such that each of the basic Σ-functions is continuous.
(b) An (N-)*standard topological partial algebra* is a topological partial algebra which is also an (N-)standard partial algebra, such that the carriers $\mathbb{B}$ (and $\mathbb{N}$) have the discrete topology.

Example 2.5.3
(a) *Discrete algebras:* The standard algebras $\mathscr{B}$ and $\mathscr{N}$ of booleans and naturals respectively (Sections 2.1 and 2.2) are topological (total) algebras under the discrete topology. All functions on them are trivially continuous, since the carriers are discrete.
(b) A *topological partial real algebra* is formed from the partial real algebra $\mathscr{R}_p$ (2.2.4(c)), or its N-standardised version $\mathscr{R}_p^N$ (2.3.3), by giving $\mathbb{R}$ its usual topology, and $\mathbb{B}$ and $\mathbb{N}$ the discrete topology. Note that the *partial* operations $\mathsf{eq}_{\mathsf{real}}^{\mathscr{R}}$ and $\mathsf{less}_{\mathsf{real}}^{\mathscr{R}}$ are continuous, in the sense of Definition 2.5.1 (whereas their total versions are not!).
(c) The N-standard *topological total real algebra* $\mathscr{R}_t^N$ is defined by

algebra	$\mathscr{R}_t^N$
import	$\mathscr{R}_0^B, \mathscr{N}$
functions	$\mathsf{div}_{\mathsf{nat}}^{\mathscr{R}} : \mathbb{R} \times \mathbb{N} \to \mathbb{R}$
end	

Here $\mathscr{R}_0^B$ is the standardised ring of reals (2.2.4(b)), $\mathscr{N}$ is the standard algebra of naturals (2.2.4(a)), and $\mathsf{div}_{\mathsf{nat}}$ is the total (continuous!) function defined by

$$\mathsf{div}_{\mathsf{nat}}(x, n) = \begin{cases} x/n & \text{if } n \neq 0, \\ 0 & \text{if } n = 0. \end{cases}$$

Note that $\mathscr{R}_t^N$ does *not* contain total boolean-valued functions "<" or "=" on the reals, since they are not continuous (cf. the partial functions $\mathsf{eq}_{\mathsf{real}}$ and $\mathsf{less}_{\mathsf{real}}$ of $\mathscr{R}_p$); nor does it contain division of reals by reals (since that cannot be total and continuous).

2.6. Metric algebra

A particular type of topological algebra is a *metric partial algebra*. This is a many-sorted standard partial algebra A with an associated metric:

algebra	A
import	$\mathscr{R}_0^B$
carriers	$A_1, \ldots A_r$
functions	$F_1^A : A^{u_1} \to A_{s_1}$,
	$\ldots,$
	$F_k^A : A^{u_k} \to A_{s_k}$,
	$\mathsf{d}_1^A : A_1^2 \to \mathbb{R}$,
	$\ldots,$
	$\mathsf{d}_r^A : A_r^2 \to \mathbb{R}$
end	

where $\mathscr{R}_0^B$ is the standardised ring of reals (Example 2.2.4(b)), the carriers $A_1, \ldots, A_r$ are metric spaces with metrics $\mathsf{d}_1^A, \ldots, \mathsf{d}_r^A$ respectively, $F_1, \ldots, F_k$ are the Σ-function symbols other than $\mathsf{d}_1, \ldots, \mathsf{d}_k$, and the (partial) functions F_i^A are all continuous with respect to these metrics. (Recall Definition 2.5.1 for the continuity of partial functions. Note that the metrics d_i^A are automatically continuous w.r.t. the topology they define.)

Clearly, metric algebras can be viewed as special cases of *topological partial algebras*. The carrier $\mathbb{B}$ (as well as $\mathbb{N}$, if present) is given the *discrete metric*, which induces the discrete topology.

Example 2.6.1. The partial and total real algebras $\mathscr{R}_p$, $\mathscr{R}_p^N$ and $\mathscr{R}_t^N$ (Examples 2.5.3) can be recast as metric algebras in an obvious way.

A topological algebra A can be expanded to a topological algebra A^* of arrays over A in a standard way. Correspondingly, a metric algebra A can be expanded to a metric algebra A^*.

3. Local uniform *While** approximations on total metric algebras, Weierstrass approximability and GL-computability

In [51, Section 9] we showed the equivalence between ***While**** approximability and Weierstrass approximability of total functions on connected total metric algebras (under certain conditions), and applied this to proving the equivalence between Grzegorczyk–Lacombe computability on the unit interval [0, 1] and ***While**** approximability on a certain total algebra $\mathscr{I}_t^N$ on [0, 1].

The aim of this section is to extend these concepts of computable approximation so as to apply to non-compact spaces such as the real line $\mathbb{R}$ instead of [0, 1]. We do this by considering *effective local* notions of uniform approximability and continuity of functions on total metric algebras. The key concept for this is that of an *open exhaustion* of a (subspace of a) metric space.

The theory in this Section, as in [51, Section 9], will be developed for *total metric algebras* in Sections 3.1 and 3.2. In Section 3.3 it will be applied to the total metric algebra $\mathscr{R}_t^N$ on $\mathbb{R}$.

3.1. Open exhaustions; global and local uniform approximability and continuity

This subsection contains the definitions of the concept of open exhaustion in a metric space, and corresponding local and global concepts of effective uniform approximability and effective uniform continuity.

Definition 3.1.1 (*Open exhaustion*). An *open exhaustion in a metric space X* is a pair (U, V) where V is a sequence of open subsets of X

$$V = (V_0, V_1, V_2, \ldots) \quad \text{and} \quad \bigcup_{p=0}^{\infty} V_p = U.$$

Remark 3.1.2

(1) If (U, V) is an open exhaustion in X, then U is open in X.

(2) If (U, V) is an open exhaustion in X, we also say that V *is an open exhaustion of* $U \subseteq X$.

(3) It is often convenient, though not necessary, to assume that the sequence V is *increasing*, i.e.,

$$V_0 \subseteq V_1 \subseteq V_2 \subseteq \cdots.$$

In any case we may assume this w.l.o.g., since (at least for the purposes considered below) we can otherwise replace (U, V) by the exhaustion (U, V'), where $V'_p = \bigcup_{i=0}^{p} V'_i$.

Example 3.1.3

(a) The *trivial exhaustion* of a metric space X is (U, V) where $U = X$ and for all p, $V_p = X$.

(b) The *standard open exhaustion* of $\mathbb{R}^q$ $(q = 1, 2, \ldots)$ is $(\mathbb{R}^q, V)$, where $V_p = (-p, p)^q$.

(c) The *standard open exhaustion* of $\mathbb{R}^+ =_{df} \{x \in \mathbb{R} \mid x > 0\}$, is $(\mathbb{R}^+, V)$, where $V_p = (1/p, p)$.

Definition 3.1.4 (*Connected exhaustion*). An open exhaustion (U, V) of X is said to be *connected* if U is connected (as a subspace of X).

We will be concerned mainly with functions f defined (at least) on U, where (U, V) is a given open exhaustion of X. We therefore introduce the following definition and notation.

Definition 3.1.5. For any $U \subseteq X$, a function $f : X \to Y$ is *total on U* if $\boldsymbol{dom}(f) \supseteq U$. We write $f : X \underset{U}{\to} Y$ to indicate that f is total on U.

Later (4.2.10) we will introduce an "effectivity condition" on open exhaustions (which holds for all the examples in 3.1.3). For now we will frame some effectivity notions with respect to arbitrary (non-effective) exhaustions.

Definition 3.1.6 (*Effective local uniform approximability*). Let X and Y be metric spaces, and let (U, V) be an open exhaustion of X. Given a function $f : X \underset{U}{\to} Y$ and a sequence of functions $g_n : X \underset{U}{\to} Y$ $(n = 0, 1, 2, \ldots)$, we say that g_n *approximates* f *effectively locally uniformly* (or *converges effectively locally uniformly to* f) w.r.t. (U, V) iff there is a recursive function $\mu : \mathbb{N}^2 \to \mathbb{N}$ such that for all p, m, n and all $x \in V_p$,

$$m \geqslant \mu(p, n) \quad \Rightarrow \quad \mathsf{d}_Y(g_m(x), f(x)) < 2^{-n}.$$

Lemma 3.1.7. *If g_n approximates f effectively locally uniformly w.r.t. (U, V), and each g_n is continuous on U, then so is f.*

Proof. This is a standard result of real analysis [39] (which holds without the assumption of recursiveness of μ), but we repeat the proof because of its importance. Let $a \in U$ and $\epsilon > 0$. Then for some p, $a \in V_p$. Choose n such that $2^{-n} \leqslant \epsilon/3$. Let $m = \mu(p, n)$ (in the notation of Definition 3.1.6). Then

$$x \in V_p \quad \Rightarrow \quad \mathsf{d}(g_m(x), f(x)) < 2^{-n} \leqslant \epsilon/3. \tag{1}$$

Since V_p is open, we can take $\delta_0 > 0$ such that

$$\mathbf{B}(a, \delta_0) \subseteq V_p, \tag{2}$$

(where $\mathbf{B}(a, \delta_0)$ is the open ball with centre a and radius δ_0). By continuity of g_m, there is $\delta > 0$ such that $\delta \leqslant \delta_0$ and

$$x \in \mathbf{B}(a, \delta) \quad \Rightarrow \quad \mathsf{d}(g_m(x), g_m(a)) < \epsilon/3. \tag{3}$$

From (1)–(3), we have, for all $x \in \mathbf{B}(a, \delta)$:

$$\begin{aligned} \mathsf{d}(f(x), f(a)) &\leqslant \mathsf{d}(f(x), g_m(x)) + \mathsf{d}(g_m(x), g_m(a)) + \mathsf{d}(g_m(a), f(a)) \\ &< \epsilon/3 + \epsilon/3 + \epsilon/3 \\ &= \epsilon. \qquad \square \end{aligned}$$

As a special case of local uniform approximability with the trivial exhaustion, we have:

Definition 3.1.8 (*Effective global uniform approximability*). Let X and Y be metric spaces. Given a total function $f : X \to Y$ and a sequence of total functions $g_n : X \to Y$ $(n = 0, 1, 2, \ldots)$, we say:

(a) g_n *approximates* f *effectively globally uniformly* (or *converges effectively globally uniformly to* f) *on* X iff there is a recursive function $\mu : \mathbb{N} \to \mathbb{N}$ such that for all m, n and all $x \in X$,

$$m \geqslant \mu(n) \quad \Rightarrow \quad \mathsf{d}_Y(g_m(x), f(x)) < 2^{-n};$$

(b) g_n *fast approximates* f *globally uniformly on* X iff for all n and all $x \in X$,

$$\mathsf{d}_Y(g_n(x), f(x)) < 2^{-n}.$$

Remark 3.1.9. Given a sequence g_n which approximates f effectively globally uniformly, we can find a subsequence which *fast approximates* f globally uniformly, namely (assuming w.l.o.g. that μ is increasing) $g'_n =_{df} g_{\mu(n)}$. Notice that although the functions g_n need not be computable, the construction of this subsequence is effective in the given sequence, since μ is recursive.

Similarly, there are "local" and "global" concepts of effective uniform continuity.

Definition 3.1.10 (*Effective local uniform continuity*). Let X and Y be metric spaces, and let (U, V) be an open exhaustion of X. We say that a function $f : X \underset{U}{\rightarrow} Y$ is *effectively locally uniformly continuous w.r.t.* (U, V), iff there is a recursive function $\delta : \mathbb{N}^2 \rightarrow \mathbb{N}$ such that for all p, n and all $x, y \in V_p$:

$$\mathsf{d}_X(x, y) < 2^{-\delta(p,n)} \quad \Rightarrow \quad \mathsf{d}_Y(f(x), f(y)) < 2^{-n}.$$

Again, as a special case of this, relative to the trivial exhaustion of X, we have:

Definition 3.1.11 (*Effective global uniform continuity*). Let X and Y be metric spaces. We say that a total function $f : X \rightarrow Y$ is *effectively globally uniformly continuous* iff there is a recursive function $\delta : \mathbb{N} \rightarrow \mathbb{N}$ such that for all n and all $x, y \in X$:

$$\mathsf{d}_X(x, y) < 2^{-\delta(n)} \quad \Rightarrow \quad \mathsf{d}_Y(f(x), f(y)) < 2^{-n}.$$

*3.2. Local uniform **While*** approximability and Weierstrass approximability*

Recall the ***While*** and ***While**** programming languages and their semantics [52]. The ***While**** language includes auxiliary array variables of sort s^* for each Σ-sort s. Most of our concepts and results come in two forms, related to the ***While*** and to the ***While**** languages. [1]

Let A be a total metric algebra of signature Σ. Let u be a Σ-product type and s a Σ-sort. Given a function procedure $P : \mathsf{nat} \times u \rightarrow s$, we write, for any $n \in \mathbb{N}$:

$$P_n^A =_{df} P^A(n, \cdot) : A^u \rightarrow A_s.$$

We turn to open exhaustions in metric algebras. Let (U, V) be an open exhaustion of A^u.

Definition 3.2.1 (*Local uniform **While**$^{(*)}$ approximability*). A function $f : A^u \underset{U}{\rightarrow} A_s$ is *locally uniformly **While**$^{(*)}$ approximable on A w.r.t.* (U, V), if there is a ***While***$^{(*)}$ (Σ^N) procedure $P : \mathsf{nat} \times u \rightarrow s$ such that

(i) for all n, the ***While***$^{(*)}$ computable function $P_n^{A^N}$ is total on U, and

(ii) the sequence of ***While***$^{(*)}$ computable functions $P_n^{A^N}$ approximates f effectively locally uniformly w.r.t. (U, V), i.e., there is a recursive function $\mu : \mathbb{N}^2 \rightarrow \mathbb{N}$ such that for all p, m, n and all $x \in V_p$,

$$m \geqslant \mu(p, n) \quad \Rightarrow \quad \mathsf{d}\Big(P_m^{A^N}(x), f(x)\Big) < 2^{-n},$$

or, in other words:

$$m \geqslant \mu(p, n) \quad \Rightarrow \quad \mathsf{d}\Big(P^{A^N}(m, x), f(x)\Big) < 2^{-n}.$$

Remark 3.2.2. An equivalent formulation of part (ii) of this definition is:
(ii′) there is a recursive function $\hat{\mu} : \mathbb{N} \rightarrow \mathbb{N}$ such that for all m, n and all $x \in V_n$,

[1] We will write "***While***$^{(*)}$" for "***While*** or ***While****".

$$m \geqslant \hat{\mu}(n) \Rightarrow \mathsf{d}\Big(P^{A^N}(m,x),\ f(x)\Big) < 2^{-n}.$$

(Just define $\hat{\mu}(n) =_{df} \mu(n,n)$, or, conversely, $\mu(p,n) =_{df} \hat{\mu}(\max(p,n))$. Further, by Remark 3.1.9, we can replace this by the simpler:

(ii″) for all n and all $x \in V_n$, $\mathsf{d}(P^{A^N}(n,x),\ f(x)) < 2^{-n}$.

Lemma 3.2.3. *If $f : A^u \underset{U}{\rightarrow} A_s$ is locally uniformly* ***While***$^{(*)}$ *approximable on A w.r.t. (U,V), then f is continuous on U.*

Proof. This follows from Lemma 3.1.7, since f is locally uniformly approximated by the functions $P_n^{A^N}$, which are continuous by the Continuity Theorem [51, Section 6], which states that all ***While***$^{(*)}$ computable functions are continuous. □

Again, as a special case of local uniform approximability with the trivial exhaustion of A^u, we have:

Definition 3.2.4 (*Global uniform* ***While***$^{(*)}$ *approximability*). A total function $f : A^u \to A_s$ is *globally uniformly* ***While***$^{(*)}$ *approximable on A* if there is a ***While***$^{(*)}$ (Σ^N) procedure $P : \mathsf{nat} \times u \to s$ such that

(i) for all n, $P_n^{A^N}$ is total on A^u, and

(ii) the sequence $P_n^{A^N}$ approximates f globally uniformly.

Remark 3.2.5. Again, by Remark 3.1.9, we can replace clause (ii) in the definition by the simpler:

(ii′) for all n and all $x \in A^u$, $\mathsf{d}(P^{A^N}(n,x),\ f(x)) < 2^{-n}$.

Next, in order to speak of effective Weierstrass approximability, i.e., effective approximability by a sequence of terms, we need some terminology in connection with the *effective representation of term evaluation*.

Let $\mathrm{x} \equiv (\mathrm{x}_1,\ldots,\mathrm{x}_n) : u$. Let $\boldsymbol{Term}_{\mathrm{x}}(\Sigma)$ be the class of all Σ-terms with variables among x only, and let $\boldsymbol{Term}_{\mathrm{x},s}(\Sigma)$ be the class of such terms of sort s. The term evaluation representing function on A relative to x and s is the function

$$\boldsymbol{te}^A_{\mathrm{x},s} : \ulcorner \boldsymbol{Term}_{\mathrm{x},s}(\Sigma) \urcorner \times A^u \to A_s$$

defined by: $\boldsymbol{te}^A_{\mathrm{x},s}(\ulcorner t \urcorner, a_1,\ldots,a_n) =$ value of t in A_s when x_i is assigned the value a_i ($i = 1,\ldots,n$), where $\ulcorner t \urcorner$ is the Gödel number of t, and if S is a set of terms, then $\ulcorner S \urcorner$ is the set of their Gödel numbers.

Definition 3.2.6 (*TEP*). The algebra A is said to have the *term evaluation property* (TEP) if for all x and s, the term evaluation representing function $\boldsymbol{te}^A_{\mathrm{x},s}$ is ***While*** computable on A^N.

Many well-known varieties (i.e., equationally axiomatisable classes of algebras) have the TEP; for example, semigroups, groups, and rings, as well as $\mathscr{R}_t^N$. For a further discussion of this property, see [51,52].

Definition 3.2.7 (*Effective local and global Weierstrass approximability*)

(a) A function $f : A^u \underset{U}{\rightarrow} A_s$ is *effectively locally Σ-Weierstrass approximable over A w.r.t.* (U, V) if, for some $\mathrm{x} : u$, there is a total recursive function

$$h : \mathbb{N} \to \ulcorner \boldsymbol{Term}_{\mathrm{x},s}(\Sigma) \urcorner$$

such that, putting $g_n(a) =_{df} \boldsymbol{te}^A_{\mathrm{x},s}(h(n), a)$, the sequence g_n approximates f effectively locally uniformly on A^u w.r.t. (U, V).

(b) A total function $f : A^u \to A_s$ is *effectively globally Σ-Weierstrass approximable over A* if, for some $\mathrm{x} : u$, there is a total recursive function

$$h : \mathbb{N} \to \ulcorner \boldsymbol{Term}_{\mathrm{x},s}(\Sigma) \urcorner$$

such that, putting $g_n(a) =_{df} \boldsymbol{te}_{\mathrm{x},s}A(h(n), a)$, the sequence g_n approximates f effectively globally uniformly on A^u.

(c) *Effective local and global Σ^*-Weierstrass approximability* are defined similarly, by replacing "Σ" by "Σ^*" and "$\boldsymbol{te}^A_{\mathrm{x},s}$" by "$\boldsymbol{te}^{A^*}_{\mathrm{x},s}$".

We can rewrite parts (a) and (b) of this definition as follows. The function $f : A^u \underset{U}{\rightarrow} A_s$ is *effectively locally* (or *globally*) *Σ-Weierstrass computable* if there is an effective infinite sequence of terms $t_0, t_1, \ldots$, all with variables contained in $\mathrm{x} : u$, such that, writing

$$g_n(a) = \text{value of } t_n \text{ when x has the value } a \in A^u,$$

the sequence g_n approximates f effectively locally (or globally, respectively) uniformly on A^u w.r.t. (U, V).

Lemma 3.2.8. *For any open exhaustion (U, V) of A^u, a function $f : A^u \underset{U}{\rightarrow} A_s$ is effectively locally Σ-Weierstrass approximable w.r.t. (U, V), iff it is effectively locally Σ^*-Weierstrass approximable w.r.t. (U, V).*

Proof. This follows from the Σ^*/Σ Conservativity Theorem [52, Section 3.15], which states that every Σ^*-term of a sort in Σ, all of whose variables are also of sorts in Σ only, can be effectively transformed to a semantically equivalent Σ-term. □

We shall therefore speak of "effective local (or global) Weierstrass approximability" to mean effective local (or global) Σ- or Σ^*-Weierstrass approximability.

We apply these ideas to approximability on $\mathbb{R}$.

Definition 3.2.9 (*$\mathbb{Q}$-polynomial definability on* $\mathbb{R}$). Let

$$\boldsymbol{Poly}^q_{\mathrm{x}} = \mathbb{Q}[\mathrm{x}_1, \ldots, \mathrm{x}_q]$$

be the set of polynomial expressions in $\mathrm{x} \equiv \mathrm{x}_1, \ldots, \mathrm{x}_q$ with rational coefficients. A function $f : \mathbb{R}^q \to \mathbb{R}$ is $\mathbb{Q}$-polynomially definable on $\mathbb{R}$ if it is explicitly definable by a term in $\boldsymbol{Poly}^q_{\mathrm{x}}$.

Lemma 3.2.10 (Equivalence of explicit and $\mathbb{Q}$-polynomial definability on $\mathbb{R}$). *A $\Sigma(\mathscr{R}^N_t)$-term of sort* real *can be effectively transformed to a semantically equivalent $\mathbb{Q}$-polynomial.*

Proof. Briefly: we eliminate all occurrences of the "if" operator in the term, using connectedness of $\mathbb{R}^q$ [51, Section 9, Lemma 2]. The result can easily be expressed as a $\mathbb{Q}$-polynomial. □

Definition 3.2.11 (*Effective local and global $\mathbb{Q}$-polynomial approximability on* $\mathbb{R}$). Let

$$\boldsymbol{val}_{\mathrm{x}q} : \ulcorner \boldsymbol{Poly}^q_{\mathrm{x}} \urcorner \times \mathbb{R}^q \to \mathbb{R}$$

be the standard evaluation of $\boldsymbol{Poly}^q_{\mathrm{x}}$ in $\mathbb{R}$.

(a) A function $f : \mathbb{R}^q \underset{U}{\to} \mathbb{R}$ is *effectively locally $\mathbb{Q}$-polynomially approximable on $\mathbb{R}$ w.r.t. an exhaustion (U, V) of $\mathbb{R}^q$* if there is a total recursive function

$$h : \mathbb{N} \to \ulcorner \boldsymbol{Poly}^q_{\mathrm{x}} \urcorner$$

such that, putting $g_n(a) =_{df} \boldsymbol{val}^q_{\mathrm{x}}(h(n), a)$, the sequence g_n approximates f effectively locally uniformly on $\mathbb{R}^q$ w.r.t. (U, V).

(b) *Effective global $\mathbb{Q}$-polynomial approximability* of a total function on $\mathbb{R}$, or on a subinterval of $\mathbb{R}$, is defined in the obvious way.

Recall the definition (3.2.7) of Weierstrass approximability.

Lemma 3.2.12 (Equivalence of Weierstrass and $\mathbb{Q}$-polynomial approximability on $\mathbb{R}$). *Effective local (or global) Weierstrass approximability over $\mathscr{R}^N_t$ corresponds to effective local (or global) $\mathbb{Q}$-polynomial approximability on $\mathbb{R}$.*

Proof. By Lemma 3.2.10. □

Example 3.2.13

(a) The functions e^x and $\sin(x)$ are effectively locally $\mathbb{Q}$-polynomially approximable w.r.t. the standard open exhaustion of $\mathbb{R}$ (Example 3.1.3(b)) by the partial sums of their Taylor expansions, but not effectively globally uniformly approximable by any sequence of polynomials, as can be seen by considering their rate of growth as $x \to \infty$.

(b) The function $\tan(x)$ is effectively locally $\mathbb{Q}$-polynomially approximable, by the partial sums of its Taylor expansion, w.r.t. the exhaustion (U, V), where

$$U = \mathbb{R} \backslash \left\{ \left(k + \frac{1}{2}\right)\pi \mid k = 0, \pm 1, \pm 2, \ldots \right\}$$

and

$$V_p = \bigcup_{k=-p}^{p} \left(\left(k - \frac{1}{2}\right)\pi + \frac{1}{p}, \left(k + \frac{1}{2}\right)\pi - \frac{1}{p} \right) \quad (p = 1, 2, \ldots).$$

Going back to the general case, we are looking for a *local* uniform version of the Theorem in [51, Section 9] (i.e., Corollary 3.2.20 below). As in [51, Section 9], we need a special condition in each direction: for

"effective local Weierstrass $\Rightarrow$ effective local uniform ***While***$^{(*)}$"

we need the TEP, and for

"effective local uniform ***While***$^{(*)}$ $\Rightarrow$ effective local Weierstrass"

we need the BCP (boolean computability property):

Definition 3.2.14 (*BCP*). A Σ-algebra A has the *boolean computability property* (*BCP*) if for any closed Σ-boolean term b, its valuation b^A (=`tt` or `ff`, by totality) can be effectively computed, or (equivalently) there is a recursive function

$$f : \ulcorner \boldsymbol{Term}_{\emptyset,\mathsf{bool}}(\Sigma) \urcorner \to \mathbb{B}$$

with $f(\ulcorner b \urcorner) = b^A$ (where $\boldsymbol{Term}_{\emptyset,\mathsf{bool}}(\Sigma)$ is the set of closed boolean Σ-terms).

Example 3.2.15 (*Counterexample for BCP*). Let A be the standard algebra of naturals $\mathscr{N}$ (Example 2.2.4(a)) expanded by some non-recursive function $f : \mathbb{N} \to \mathbb{N}$. Then A does not have BCP, since otherwise there would be an algorithm for f at input n by testing in turn the booleans $\mathsf{f}(\bar{n}) = \bar{0}$, $\mathsf{f}(\bar{n}) = \bar{1}$, $\mathsf{f}(\bar{n}) = \bar{2}, \ldots$ (where $\bar{n}$ is the numeral for n).

On a more positive note:

Example 3.2.16. $\mathscr{R}_t^N$ has both the TEP and the BCP.

In the following lemma, A need not be a metric algebra (cf. [51, Section 9, Lemma 4]).

Lemma 3.2.17. *Suppose A has the TEP. Given variables* $\mathtt{x} : u$, *let*

$$h : \mathbb{N} \to \ulcorner \boldsymbol{Term}_{\mathtt{x},s}(\Sigma) \urcorner$$

be a total recursive function. Then there is a **While**(Σ^N) *procedure* $P : \mathsf{nat} \times u \to s$ *such that for all* $a \in A^u$ *and* $n \in \mathbb{N}$,

$$P^{A^N}(n, a) = \boldsymbol{te}^A_{\mathtt{x},s}(h(n), a).$$

For the converse direction, we need a "local" version of Lemma 5 in [51, Section 9]:

Lemma 3.2.18. *Suppose A is a total metric algebra with the BCP, and U is a connected subset of* A^u. *Let* $P : \mathsf{nat} \times u \to s$ *be a* **While**$^{(*)}$ *procedure over* A^N *which defines a function*

$$P^{A^N} : \mathbb{N} \times A^u \underset{U}{\to} A_s.$$

Then there is a total recursive function $h : \mathbb{N} \to \ulcorner \boldsymbol{Term}_{\mathtt{x},s}(\Sigma) \urcorner$ *such that for all* $a \in U$ *and* $n \in \mathbb{N}$,

$$\boldsymbol{te}^A_{\mathtt{x},s}(h(n), a) = P^{A^N}(n, a).$$

The proof uses (i) *connectedness* of U and *totality* of A to show that any boolean test gives a constant value (true or false) independent of the state (i.e., assignment of values a to $\mathtt{x}$), and (ii) the BCP to effectively decide such a test by evaluating any closed instance of the boolean term (which exists by the Instantiation Assumption).

We now have a local version of the Theorem in [51, Section 9].

Theorem 3.2.19 (Local uniform approximability: equivalent versions). *Suppose A is a total metric algebra with the TEP and BCP, and* (U, V) *is a connected open exhaustion of* A^u. *Let* $f : A^u \underset{U}{\to} A_s$. *Then the following are equivalent*:

(i) *f is effectively locally uniformly* **While** *approximable on A w.r.t.* (U, V);
(ii) *f is effectively locally uniformly* **While*** *approximable on A w.r.t.* (U, V);
(iii) *f is effectively locally Weierstrass approximable on A w.r.t.* (U, V).

Proof. From Lemmas 3.2.17 and 3.2.18. □

As a special case, using the trivial exhaustion (Example 3.1.3(a)), we derive the "global uniformity" theorem in [51, Section 9]:

Corollary 3.2.20 (Global uniform approximability: equivalent versions). *Suppose A is a total metric algebra with the TEP and BCP, and A^u is connected. Let $f : A^u \to A_s$ be a total function. Then the following are equivalent*:

(i) *f is globally uniformly* ***While*** *approximable on A*;
(ii) *f is globally uniformly* ***While***** *approximable on A*;
(iii) *f is effectively Weierstrass approximable on A.*

3.3. Application to computability on $\mathbb{R}$: GL-computability

Now we concentrate on computability on the metric space $\mathbb{R}$. We introduce a concrete model of computability on $\mathbb{R}$: Grzegorczyk–Lacombe (GL) computability, and compare it to the model of the last subsection, local uniform ***While***$^{(*)}$ approximability, applied to the total metric algebra $\mathscr{R}_t^N$. We show that these are equivalent, under the assumption of *effective local uniform continuity*.

Definition 3.3.1 (*GL-computability*)

(1) A function $f : \mathbb{R}^q \underset{U}{\to} \mathbb{R}$ is *GL* (*Grzegorczyk/Lacombe*) *computable on* $\mathbb{R}$ w.r.t. (U, V), if:
  (i) f is *sequentially computable* on U, i.e., f maps every computable sequence of points in U into a computable sequence of points in $\mathbb{R}$;
  (ii) f is *effectively locally uniformly continuous* w.r.t. (U, V).
(2) A total function $f : [0, 1]^q \to \mathbb{R}$ is *GL computable on* $[0, 1]^q$ if:
  (i) f is *sequentially computable* on $[0, 1]^q$;
  (ii) f is *effectively globally uniformly continuous* on $[0, 1]^q$.

GL-computability was developed in [14,15,24]. Our definition follows the version given in [35]. Note that it is an example of a concrete model, since it depends on the representation of computable numbers.

In [51,53] we studied the class of functions (total and) GL computable on $[0, 1]^q$. In this paper we study the functions GL computable on $\mathbb{R}^q$, with respect to the *standard open exhaustion* of $\mathbb{R}$ (Example 3.1.3(b)). Note that below, in connection with computing on $\mathbb{R}$, "local" concepts (such as "local uniform computability" and "local uniform continuity") will refer to this exhaustion.

The following theorem uses an adaptation of the argument for the Theorem in Section 10 of [51] (which applies to I) to the whole real line.

Theorem 3.3.2. *Let $f : \mathbb{R}^q \to \mathbb{R}$ be total. Then*, *w.r.t. the standard exhaustion of* $\mathbb{R}$, *the following are equivalent*:

(i) *f is locally uniformly* ***While***$(\mathscr{R}_t^N)$ *approximable on* $\mathbb{R}$;
(ii) *f is locally uniformly* ***While***$^*(\mathbb{R}_t^N)$ *approximable on* $\mathbb{R}$;
(iii) *f is effectively locally $\mathbb{Q}$-polynomially approximable on* $\mathbb{R}$;
(iv) *f is GL-computable on* $\mathbb{R}$.

Proof. Since $\mathbb{R}^q$ is connected, and $\mathscr{R}_t^N$ has the TEP and BCP, the equivalence of the first three assertions is a special case of Theorem 3.2.19, since on $\mathbb{R}$ (by Lemma 3.2.12) effective local Weierstrass approximability means effective local polynomial approximability. The equivalence of (iii) and (iv) is stated in [35] and proved in detail for the simpler case with I instead of $\mathbb{R}$. The extension to the present case is routine. (Cf. [35, Ch. 0, Theorem 6].) Note, in this connection, that effective local polynomial approximability on $\mathbb{R}$ implies effective local uniform continuity of f (part of the definition of GL-computability). □

Remark 3.3.3

(a) The historical remark given in [51, Section 10] for the Theorem there, detailing the work in [34,40], applies just as well to the present theorem.

(b) From this theorem it follows that either local uniform ***While***$^{(*)}$ computability on $\mathscr{R}_t^N$, or effective local $\mathbb{Q}$-polynomial approximability on $\mathbb{R}$ (i.e., any one of conditions (i), (ii) or (iii)) implies effective local uniform continuity on $\mathbb{R}$, this being part of the definition of GL-computability. In [35] this is proved directly for condition (iii), as part of the proof of (iii) $\Rightarrow$ (iv).

(c) In this theorem we restrict our attention to total functions and the standard exhaustion of $\mathbb{R}^q$, since that is how the proof of the equivalence (iii) $\Leftrightarrow$ (iv) is given (essentially) in [35]. We conjecture this equivalence, and hence this theorem, is true more generally, for functions $f : \mathbb{R}^q \underset{U}{\rightarrow} \mathbb{R}$ and open exhaustions (U, V) which are connected and for which the relation $\{(x, p) \in \mathbb{R} \times \mathbb{N} \mid x \in V_p\}$ is α_0-semicomputable. (Cf. Definition 4.2.8(c); here α_0 is some standard enumeration of the rationals.)

4. *WhileCC** approximability on partial metric algebras and tracking computability

In Section 3 we compared an abstract model (effective local ***While**** approximability on the total metric algebra $\mathscr{R}_t^N$ of reals) and a concrete model (Gzregorczyk–Lacombe), of computability on $\mathbb{R}$, which were shown to be equivalent for total functions. Here we consider another abstract model (***WhileCC**** approximability on metric partial algebras) in Section 4.1, and another concrete model (computability on metric partial algebras via enumerations) in Section 4.2. We prove these equivalent for partial functions f on such algebras under a number of general assumptions, including effective local uniform continuity of f (Theorem 4.2.13). Then in Section 4.3 we apply this equivalence result to computation on $\mathbb{R}$, using the *partial real metric algebra* $\mathscr{R}_p^N$. Finally, in Section 4.4, we connect these models of computation on $\mathbb{R}$ with the models of Section 3, thus proving the equivalence of the two abstract models $\mathscr{R}_t^N$ and $\mathscr{R}_p^N$, for total effectively locally uniformly continuous functions on $\mathbb{R}$.

4.1. *WhileCC** *approximability*

We recall the definition of ***WhileCC****(Σ) computability (***While****(Σ) with "countable choice") and ***WhileCC****(Σ) approximability given in [55, Section 3].

To briefly review the ***WhileCC*** (and ***WhileCC****) programming languages: as defined in [55], they are like the ***While*** (and ***While****) languages, with an extra "choose" rule of program term formation:

choose z : b,

where z is a variable of sort nat and b is a boolean term. In this paper we give a slightly more general syntax for ***WhileCC***, namely:

- *term formation* is defined without any reference to the "choose" operator, but
- *assignment* is defined more generally, by the three cases:
  - (i) $\mathrm{x} := t$ (simultaneous assignment),
  - (ii) $\mathrm{x} := \mathsf{choose}\ \mathrm{z} : b(\mathrm{z}, \ldots)$,
  - (iii) $\mathrm{x} := \mathsf{choose}\ \mathrm{z} : P(\mathrm{z}, \ldots)$,

where z is a variable of sort nat, and in (ii) $b(\mathrm{z}, \ldots)$ is a *boolean term*, and in (iii) $P(\mathrm{z}, \ldots)$ is a *semicomputable predicate* of z (and other variables), i.e., the halting set of a ***WhileCC*** function procedure with z among its input variables.

Remark 4.1.1. As is easily seen, cases (i) and (ii) alone give a programming language equivalent to that in [55]. The new case (iii) extends this language. This permits us to derive the ***WhileCC***** Archimedean property, used in the proof of the Completeness Theorem 4.2.13, from the assumption of ***WhileCC*****-semicomputability (cf. Remark 4.2.11 below).

In [55] an algebraic operational semantics is given for ***WhileCC***, whereby a ***WhileCC*** procedure $P : u \to v$ has a meaning in an N-standard Σ-algebra A:

$$P^A : A^u \to \mathscr{P}_\omega^+(A^v \cup \{\uparrow\}), \tag{4}$$

where $\mathscr{P}_\omega^+(X)$ is the set of all countable *non-empty* subsets of X, and "$\uparrow$" represents a divergent computation.

Note that the semantic definition in [55] can be adapted without difficulty to the version of ***WhileCC*** as defined above (i.e., including case (iii)), leading again to a semantics for procedures as in (4).

Now let A be a metric partial Σ-algebra, and let U be an open subset of A^u.

Definition 4.1.2 (*Uniform* ***WhileCC***** *approximability*). A function $f : A^u \underset{U}{\to} A_s$ is *uniformly approximable* on U by a ***WhileCC*****(Σ) procedure $P : \mathsf{nat} \times u \to s$ if for all $n \in \mathbb{N}$ and all $a \in A^u$:

$$a \in U \quad \Rightarrow \quad \uparrow \notin P^A(n, a) \subseteq \mathbf{B}(f(a), 2^{-n}),$$

where $\mathbf{B}(a, \delta)$ is the open ball with centre a and radius δ.

Note that the concept of uniform ***WhileCC***** approximability, unlike that of local uniform ***While***** approximability, does not refer explicitly to an exhaustion (U, V), only to the set $U \subseteq A^u$.

4.2. Metric algebras with enumerations; tracking computability

Let A be an N-standard metric Σ-algebra. Let X be a family $\langle X_s \mid s \in \boldsymbol{Sort}(\Sigma)\rangle$ of subsets $X_s \subseteq A_s$. Each X_s can be viewed as a *metric subspace* of the metric space A_s.

Definition 4.2.1. An *enumeration* of X is a family

$$\alpha = \langle \alpha_s : \Omega_s \twoheadrightarrow X_s \mid s \in \boldsymbol{Sort}(\Sigma)\rangle$$

of surjective maps $\alpha_s : \Omega_s \twoheadrightarrow X_s$, for some family

$$\Omega = \langle \Omega_s \mid s \in \boldsymbol{Sort}(\Sigma)\rangle$$

of sets $\Omega_s \subseteq \mathbb{N}$. The family X is said to be *enumerated by* α. We say that $\alpha : \Omega \twoheadrightarrow X$ is an *enumeration* of X, and call the pair (X, α) an *enumerated* ***Sort***(Σ)*-family of subspaces* of A. (The notation "$\twoheadrightarrow$" denotes surjections, or onto mappings.)

We also write $\Omega_s = \Omega_{\alpha,s}$ to make explicit the fact that $\Omega_s = \boldsymbol{dom}(\alpha_s)$, and we use the notation $\Omega^u_\alpha = \Omega_{\alpha,s_1} \times \cdots \times \Omega_{\alpha,s_m}$ and $X^u = X_{s_1} \times \cdots \times X_{s_m}$ where $u = s_1 \times \cdots \times s_m$.

Assume now that A is an N-standard metric Σ-algebra and (X, α) is an enumerated ***Sort***(Σ)-family of subspaces of A, with enumeration $\alpha : \Omega \twoheadrightarrow X$.

Definition 4.2.2 (*Tracking functions*)

(a) Let $f : A^u \overset{\cdot}{\to} A_s$ and $\varphi : \mathbb{N}^m \overset{\cdot}{\to} \mathbb{N}$. Then φ is a *strict α-tracking function for f* if the following diagram commutes:

$$\begin{tikzcd}
A^u \arrow[r, "f", "\cdot"'] & A_s \\
\Omega^u_\alpha \arrow[u, "\alpha^u"] \arrow[r, "\cdot", "\varphi"'] & \Omega_{\alpha,s} \arrow[u, "\alpha_s"']
\end{tikzcd}$$

in the sense that for all $k = (k_1, \ldots, k_m) \in \Omega^u_\alpha$, and writing $\alpha^u(k) = (\alpha_{s_1}(k_1), \ldots, \alpha_{s_m}(k_m))$:

$$\begin{aligned}
&\varphi(k)\downarrow \quad\Rightarrow\quad \varphi(k) \in \Omega_{\alpha,s} \wedge f(\alpha^u(k)) \downarrow \alpha_s(\varphi(k)), && (5)\\
&\varphi(k)\uparrow \quad\Rightarrow\quad f(\alpha^u(k))\uparrow .
\end{aligned}$$

(b) Suppose $\boldsymbol{dom}(f) \supseteq U$, i.e., $f : A^u \underset{U}{\to} A_s$. Then φ is an *α-tracking function for f on U* if for all $k \in \Omega^u_\alpha$ such that $\alpha^u(k) \in U$,

$$\varphi(k) \downarrow\in \Omega_{\alpha,s} \wedge f(\alpha^u(k)) \downarrow \alpha_s(\varphi(k)).$$

Definition 4.2.3 (*α-computability*)

(a) The function $f : A^u \overset{\cdot}{\to} A_s$ is *strictly α-computable* if it has a (partial) recursive strict α-tracking function.

(b) The function $f : A^u \underset{U}{\to} A_s$ is *α-computable on U* if it has a recursive α-tracking function on U.

Remark 4.2.4

(a) Note that (5) implies that

$$f{\restriction}X^u : X^u \overset{\cdot}{\to} X_s$$

and

$$\varphi{\restriction}\Omega^u_\alpha : \Omega^u_\alpha \overset{\cdot}{\to} \Omega_{\alpha,s}.$$

(b) In the situation of Definition 4.2.3, we are not concerned with the behaviour of φ off Ω^u_α, or of f off X^u, or (in the case of part (b)) of f off U.

We now consider the case where X is a subalgebra of A. This means that all the basic Σ-functions of X, including the metrics, are retracts of the corresponding functions of A.

Definition 4.2.5 (*Enumerated Σ-subalgebra*). Let X be a Σ-subalgebra of A. An enumeration α of X, together with a family of tracking functions for its operations, is called an *enumerated Σ-subalgebra of A*.

Definition 4.2.6 (*Σ-effective enumeration*). The enumeration α is said to be *strictly Σ-effective* if all the basic Σ-functions on A (including the metrics) are strictly α-computable.

Discussion 4.2.7 (*Computational closure*). Let X be a subspace of A, enumerated by α. We define a family

$$\mathsf{C}_\alpha(X) = \langle \mathsf{C}_\alpha(X)_s \mid s \in \boldsymbol{Sort}(\Sigma)\rangle$$

of sets $\mathsf{C}_\alpha(X)_s$ of *α-computable elements of A_s*, i.e., limits in A_s of effectively convergent Cauchy sequences (to be defined below) of elements of X_s, so that

$$X_s \subseteq \mathsf{C}_\alpha(X)_s \subseteq A_s,$$

with corresponding enumerations

$$\bar{\alpha}_s : \Omega_{\bar{\alpha},s} \twoheadrightarrow \mathsf{C}_\alpha(X)_s.$$

Writing $\bar{\alpha} = \langle \bar{\alpha}_s \mid s \in \boldsymbol{Sort}(\Sigma)\rangle$, we call the enumerated subspace $(\mathsf{C}_\alpha(X), \bar{\alpha})$ the *computable closure* of (X, α) in A.

The sets $\Omega_{\bar{\alpha},s} \subseteq \mathbb{N}$ consist of *codes* for $\mathsf{C}_\alpha(X)_s$ (w.r.t. α), i.e., pairs of numbers $c = \langle e, m\rangle$ where

(i) e is an index for a total recursive function defining the function $\alpha \circ \{e\} : \mathbb{N} \to X_s$, i.e., the sequence

$$\alpha_s(\{e\}(0)), \alpha_s(\{e\}(1)), \alpha_s(\{e\}(2)), \ldots, \tag{6}$$

of elements of X_s, and

(ii) m is an index for a modulus of convergence for this sequence:

$$\forall k, l \geqslant \{m\}(n) : \mathsf{d}_i\ (\alpha(\{e\}(k)), \alpha(\{e\}(l))) < 2^{-n}.$$

For any such code $c = \langle e, m\rangle \in \Omega_{\bar{\alpha},s}$, $\bar{\alpha}_s(c)$ is defined as the limit in A_s of the Cauchy sequence (6), and so $\mathsf{C}_\alpha(X)_s$ is the range of $\bar{\alpha}_s$:

$$\begin{tikzcd} X_s \arrow[r, phantom, "\subseteq"] & \mathsf{C}_\alpha(X)_s \arrow[r, phantom, "\subseteq"] & A. \\ \Omega_{\alpha,s} \arrow[u, "\alpha_s"] & \Omega_{\bar{\alpha},s} \arrow[u, "\bar{\alpha}_s"] & \end{tikzcd}$$

As explained in [55] (and cf. Remark 3.1.9), we get an equivalent theory if we assume that the sequences (6) are *fast* Cauchy sequences, i.e., the moduli of convergence are always the identity function on $\mathbb{N}$, and so work with "e-codes" instead of "c-codes" as elements of $\Omega_{\bar{\alpha}}$.

We are generally interested in $\bar{\alpha}$-computable (rather than α-computable) functions on A as our model of *concrete computability* on A. The best known non-trivial example of an enumerated subspace and its extension to a subspace of α-computable elements, is the subspace of rationals $\mathbb{Q} \subset \mathbb{R}$ and its extension to the *recursive reals*, considered in Section 4.3.

Fix a Σ-product type u. We consider notions of semicomputability on A^u corresponding to both models of computability.

Definition 4.2.8 (*Notions of semicomputability*)
(a) The *halting set* of a ***WhileCC****** procedure $P : u \to v$ on A is the set

$$\{a \in A^u \mid P^A(a)\backslash\{\uparrow\} \neq \emptyset\}.$$

(b) A subset of A^u is ***WhileCC****-*semicomputable* if it is the halting set of some ***WhileCC**** procedure.
(c) A set $U \subseteq A^u$ is $\bar{\alpha}$-*semicomputable* if there is an r.e. relation $S \subseteq \mathbb{N}^m$ such that

$$\bar{\alpha}^{-1}[U] =_{df} \{c \in \Omega^u_{\bar{\alpha}} \mid \bar{\alpha}(c) \downarrow\in U\} = S \cap \Omega^u_{\bar{\alpha}}.$$

Lemma 4.2.9 (Domains of definition)
(a) *Suppose U is **WhileCC****-semicomputable. If $f : A^u \underset{U}{\to} A_s$ is uniformly **WhileCC**** approximable on U, then it is approximable by a procedure $P : \mathsf{nat} \times u \to s$ which diverges off U, i.e.* (*cf. Definition* 4.1.2),

$$a \in A^u\backslash U \Rightarrow P^A(n, a) = \{\uparrow\}.$$

(b) *Suppose U is $\bar{\alpha}$-semicomputable. If $f : A^u \underset{U}{\to} A_s$ is $\bar{\alpha}$-computable on U, then it is $\bar{\alpha}$-computable by a recursive tracking function φ which diverges on $\Omega^u_{\bar{\alpha}}\backslash(\bar{\alpha})^{-1}[U]$, i.e., for all $c \in \Omega^u_{\bar{\alpha}}$,*

$$\varphi(c) \downarrow \iff \bar{\alpha}(c) \in U.$$

Proof. The proofs use standard computability-theoretic arguments. For part (b), for example: by assumption, $\bar{\alpha}^{-1}[U] = S \cap \Omega^u_{\bar{\alpha}}$ for some r.e. S, and f has a recursive α-tracking function φ' on U. We modify φ' to a recursive tracking function φ as desired, with the following algorithm: "With input k, generate the elements of S until k appears. Then compute $\varphi'(k)$". Part (a) is handled similarly. □

We now consider computability concepts of open exhaustions.

Definition 4.2.10 (***WhileCC**** *computability concepts of open exhaustions*). An open exhaustion (U, V) of A^u is said to be
(a) ***WhileCC****-*semicomputable* if the relation

$$\mathsf{Loc}^A(a, p) =_{df} a \in V_p$$

is ***WhileCC****-semicomputable;
(b) ***WhileCC****-*computably open* if there is a ***WhileCC****-computable function

$$\gamma : A_u \times \mathbb{N} \to \mathbb{N}$$

such that for all p and all $a \in V_p$,

$$\mathbf{B}(a, 2^{-\gamma(a,p)}) \subseteq V_p.$$

Remark 4.2.11 (***WhileCC****-*computable Archimedean property of open exhaustions*). If (U, V) is ***WhileCC****-semicomputable, then there is a ***WhileCC**** procedure $P_{\mathsf{loc}} : u \to \mathsf{nat}$ which, given $a \in A^u$, produces some p such that $a \in V_p$, i.e.,

$$P^A_{\mathsf{loc}}(a) = \begin{cases} \{p \mid a \in V_p\} & \text{if } a \in U, \\ \{\uparrow\} & \text{otherwise.} \end{cases}$$

This procedure can be simply defined as

$$P_{\mathsf{loc}}(\mathtt{x}) ::= \mathsf{choose}\,\mathtt{p} : \mathsf{Loc}(\mathtt{x}, \mathtt{p}).$$

This ***WhileCC****-computable Archimedean property of (U, V) was used explicitly as an assumption in the Completeness Theorem C in [55]. In the present formulation of this theorem (4.2.13), this assumption is replaced by the assumption of ***WhileCC****-semicomputability of (U, V), since it can be derived from the latter, as shown above. This was not possible in [55] with its more restrictive version of the "choose" construct (cf. Remark 4.1.1).

Lemma 4.2.12. *If (U, V) is **WhileCC****-semicomputable, then so is U.*

Proof. U is the halting set of P_{loc} (in the notation of the Remark 4.2.11). □

The next theorem is a mild generalisation of the Completeness Theorem (Theorem C) in [55] (see Remark 4.2.14 below).

Theorem 4.2.13 (Completeness: equivalence of abstract and concrete models). *Let A be an N-standard partial metric Σ-algebra, with an enumerated **Sort**(Σ)-subspace (X, α). Suppose the enumerated **Sort**(Σ)-space $(\mathsf{C}_\alpha(X), \bar{\alpha})$ of α-computable elements of A is a Σ-subalgebra of A. Assume also that for all Σ-sorts s,*

(i) *$\bar{\alpha}$ is strictly Σ-effective,*
(ii) *X_s is dense in A_s, and*
(iii) *$\alpha_s : \mathbb{N} \to A_s$ is **WhileCC****-computable on A.*

For a given Σ-sort s and Σ-product type u, let (U, V) be an open exhaustion in A^u and let $f : A^u \underset{U}{\rightarrow} A_s$ be a function on A such that

(iv) *(U, V) is **WhileCC****-semicomputable and **WhileCC****-computably open, and*
(v) *f is effectively locally uniformly continuous w.r.t. (U, V).*

Then the following are equivalent:

(1) *f is uniformly **WhileCC****(Σ) approximable on U,*
(2) *f is $\bar{\alpha}$-computable on U.*

Proof. (Outline): This follows along the lines of the proof of the Completeness Theorem C in [55]. The *soundness* direction (1) ⇒ (2) constructs an $\bar{\alpha}$-tracking function for f on U from a uniform sequence of $\bar{\alpha}$-tracking functions for the ***WhileCC***-approximations to f. This uses assumption (i), i.e., Σ-effectiveness of $\bar{\alpha}$. (A deterministic version of this direction, i.e., without "choose", was proved in [47].)

The *adequacy* direction (2) ⇒ (1) uses assumptions (ii)–(v). The following is an informal overview of this direction. (See Fig. 1.)

Given the assumptions (ii)–(iv) of the theorem, suppose $f : A^u \underset{U}{\rightarrow} A_s$ is $\bar{\alpha}$-computable on U by $\varphi : \Omega^u_{\bar{\alpha}} \overset{\cdot}{\rightarrow} \Omega_{\bar{\alpha},s}$. (In the figure, we represent φ as mapping Ω^u_α to $\Omega_{\alpha,s}$, rather than $\Omega^u_{\bar{\alpha}}$ to $\Omega_{\bar{\alpha},s}$, as a convenient simplification.) We must describe a ***WhileCC****(Σ) procedure which approximates f on A.

Let $x \in U$, and suppose $f(x) \downarrow y$. By the *density* of $X_u = \boldsymbol{ran}(\alpha^u)$ in A^u, and by the *openness* of U, for each n we can find (using the "choose" operator, as well as the

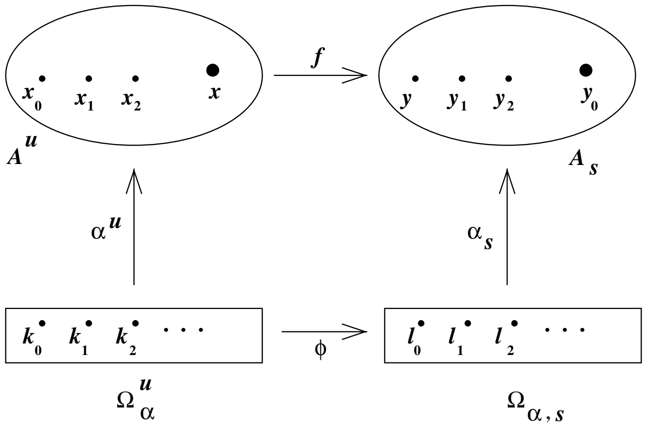

Fig. 1.

WhileCC*** computability of α) an element k_n of Ω^u_α such that $x_n =_{df} \alpha^u(k_n) \in U$, and also $\mathsf{d}(x_n, x) < 2^{-n}$.

Now compute an element l_n of Ω^u_α which is a close approximation to $f(k_n)$, or rather to $f(\mathsf{const}(k_n))$, where $\mathsf{const}(k)$ is a standard index for the constant function on $\mathbb{N}$ with value k. More precisely, let $e'_n =_{df} f(\mathsf{const}(k_n))$, and let $l_n =_{df} \{e'_n\}(n)$. Then $\mathsf{d}(\alpha(l_n), \bar{\alpha}(e'_n)) < 2^{-n}$. Put $y_n = \alpha(l_n)$. We must now check that the mapping $(x, n) \mapsto y_n$ defined above is ***WhileCC**** computable, and approximates f. By assumption (v): *effective local uniform continuity of* f w.r.t. (U, V), since $(x_n)_n$ is a fast Cauchy sequence with limit x, $(y_n)_n$ is a Cauchy sequence with limit y and ***WhileCC**** computable modulus of convergence. In fact we can assume w.l.o.g. that $\mathsf{d}(y_n, y) < 2^{-n}$ (cf. Remark 3.1.9). Note that ***WhileCC**** computability of y_n (as a function of x and n) uses the ***WhileCC**** computability of α. Hence we can define a ***WhileCC**** procedure $P : \mathsf{nat} \times u \to s$ with $P^A(n, x)$ equal to the set of all such y_n, obtainable in this way from all possible implementations of the "choose" operator. Hence f is computably approximable by P. $\square$

Remark 4.2.14 (*Connection with Completeness Theorem of [55]*). There are two slight differences between the formulations of the Completeness Theorem here and in [55, Theorem C]:

(1) In [55], assumption (iv) includes the assumption of the ***WhileCC****-computable Archimedean property of the exhaustion (U, V). In the present formulation, this assumption is replaced by (and derived from) the assumption of ***WhileCC****-semicomputability of (U, V), as explained in Remark 4.2.11.

(2) The present formulation, with $U \supseteq \boldsymbol{dom}(f)$, apparently generalises that in [55], where it was assumed that $U = \boldsymbol{dom}(f)$. However the present formulation could in any case be derived from the version in [55], by replacing f by its restriction to U.

From the Completeness Theorem 4.2.13, together with Lemmas 4.2.12 and 4.2.9, we infer:

Corollary 4.2.15 (Corollary to Completeness Theorem). *Under the assumptions of Theorem* 4.2.13, *including* (i)–(v), *together with*
(vi) *U is $\bar{\alpha}$-semicomputable*,
and assuming either (1) *or* (2), *we may conclude*:
(1^+) *f is uniformly* ***WhileCC****$^*(\Sigma)$ approximable on U by a procedure which diverges off U, and*
(2^+) *f is $\bar{\alpha}$-computable on U by a partial recursive tracking function which diverges off $(\bar{\alpha})^{-1}[U]$.*

4.3. Application to computability on $\mathbb{R}$

Let

$$\alpha_0 : \mathbb{N} \to \mathbb{Q}$$

be a (fixed, standard) enumeration of the rationals. From this we construct the set

$$\mathsf{C}_0 = \mathsf{C}_{\alpha_0}(\mathbb{Q})$$

of *recursive reals*, with enumerations

$$\bar{\alpha}_0 : \Omega_0 \twoheadrightarrow \mathsf{C}_0.$$

Note that α_0 is ***While***** computable over $\mathscr{R}_p^N$. Further, $\mathbb{Q}$ is dense in $\mathbb{R}$, C_0 is a subfield of $\mathbb{R}$, and $\bar{\alpha}_0$ is $\Sigma(\mathscr{R}_p^N)$-effective. We then have, as a corollary to Theorem 4.2.13:

Theorem 4.3.1. *Suppose $f : \mathbb{R}^q \underset{U}{\to} \mathbb{R}$ is effectively locally uniformly continuous w.r.t. (U, V), where (U, V) is a* ***WhileCC****$^*(\mathscr{R}_p^N)$-semicomputable open exhaustion of $\mathbb{R}^q$. Then the following are equivalent*:
(i) *f is $\bar{\alpha}_0$-computable on U*
(ii) *f is uniformly* ***WhileCC****$^*(\mathscr{R}_p^N)$ approximable on U.*

Here and below we use notation such as "***While****$^*(A)$" for the ***While****$^*(\Sigma(A))$ programming language, etc.

Note that Theorem 4.3.1 applies in particular to total functions on $\mathbb{R}^q$ which are effectively locally uniformly continuous w.r.t. the standard open exhaustion of $\mathbb{R}^q$.

4.4. Equivalence of all the models for total functions on $\mathbb{R}$

We connect all the models considered so far in Sections 3 and 4 for total functions on $\mathbb{R}$.

Theorem 4.4.1. *Suppose $f : \mathbb{R}^q \to \mathbb{R}$ is total and effectively locally uniformly continuous. Then the following are equivalent*:
(i) *f is GL-computable on $\mathbb{R}$,*
(ii) *f is $\bar{\alpha}_0$-computable on $\mathbb{R}$,*
(iii) *f is effectively locally $\mathbb{Q}$-polynomially approximable on $\mathbb{R}$,*
(iv) *f is locally uniformly* ***While****$(\mathscr{R}_t^N)$ approximable on $\mathbb{R}$,*
(v) *f is locally uniformly* ***While****$^*(\mathscr{R}_t^N)$ approximable on $\mathbb{R}$,*
(vi) *f is uniformly* ***WhileCC****$^*(\mathscr{R}_p^N)$ approximable on $\mathbb{R}$.*

Note that the local uniform continuity and approximability in the statement all refer to the standard open exhaustion of $\mathbb{R}^q$.

Note also that in comparing (v) with (vi), *local* uniform $\boldsymbol{While}^*(\mathscr{R}_t^N)$ approximability corresponds to *global* uniform $\boldsymbol{WhileCC}^*(\mathscr{R}_p^N)$ approximability.

Proof. (Outline). The equivalence of (i), (iii), (iv) and (v) is Theorem 3.3.2. The equivalence of (ii) and (vi) is Theorem 4.3.1. Finally, we can show the equivalence of the two concrete models (i) and (ii). We omit details. □

Remark 4.4.2. We state this theorem for total functions, and the standard exhaustion of $\mathbb{R}^q$, since those were the assumptions in Theorem 3.3.2. We conjecture the theorem is true more generally (cf. Remark 3.3.3).

Next, by working with the corresponding models over the unit interval [0, 1] with the trivial exhaustion, we obtain:

Theorem 4.4.3. *Suppose $f : [0, 1]^q \to \mathbb{R}$ is total and effectively uniformly continuous. Then the following are equivalent*:

(i) *f is GL-computable on* [0, 1],

(ii) *f is $\bar{\alpha}_0$-computable on* [0, 1],

(iii) *f is effectively $\mathbb{Q}$-polynomially approximable on* [0, 1],

(iv) *f is uniformly* $\boldsymbol{While}(\mathscr{I}_t^N)$ *approximable on* [0, 1],

(v) *f is uniformly* $\boldsymbol{While}^*(\mathscr{I}_t^N)$ *approximable on* [0, 1],

(vi) *f is uniformly* $\boldsymbol{WhileCC}^*(\mathscr{I}_p^N)$ *approximable on* [0, 1].

Here α_0 is a standard effective enumeration of the rationals between 0 and 1, and $\mathscr{I}_t^N$ and $\mathscr{I}_p^N$ are algebras on [0, 1] defined analogously to $\mathscr{R}_t^N$ and $\mathscr{R}_p^N$ respectively.

Note that this is a re-formulation of the Theorem in [51, Section 10], with the added assumption of effective uniform continuity of f, and more equivalences ((ii) and (vi)) added.

5. Algebraic specifications of GL-computable functions on the reals

In this section we modify the theory of [53, Section 6] for compact intervals to obtain a universal algebraic specification for computably approximable functions on $\mathbb{R}$, from which we obtain a universal specification for dynamical systems on Euclidean n-space.

5.1. Universal specification of *While**-approximable functions on metric algebras

We begin with a universal algebraic specification over N-standard metric algebras. This specification consists of finitely many algebraic formulae over the signature that are "localised" with respect to an exhaustion.

Let Σ be an N-standard signature, $u \to s$ a Σ-function type, and A a total N-standard metric Σ-algebra.

Definition 5.1.1 (*Conditional localised equations and inequalities*)

(a) *Conditional Σ-equations* are formulae of the form

$$P_1 \wedge \cdots \wedge P_n \to P, \tag{7}$$

where $n \geqslant 0$ and the atoms P_i and P are *equations* $(t_1 = t_2)$ where t_1 and t_2 have the same Σ-type (for any Σ-type).

(b) *Conditional Σ-equations and inequalities* are as in (a), except that the atoms in (7) may *also* be *real inequalities* $(t_1 < t_2)$ where $t_1 : \mathsf{real}$ and $t_2 : \mathsf{real}$.

(c) *Conditional localised Σ-equations and inequalities w.r.t. $u \to s$* are as in (b), except that the atoms in (7) may *also* be *localising atoms* $\mathsf{Loc}(t_1, t_2)$ (meaning "$t_1 \in V_{t_2}$") where $t_1 : u$ and $t_2 : \mathsf{nat}$.

Note that clause (c) in the definition extends the concept of "conditional equations and inequalities", introduced in [53], by the addition of "localising atoms".

Our *specification language* for functions on exhaustions is based on conditional localised equations and inequalities. The equality operator "=" and inequality operator "<" on the reals, and the locality predicate "Loc" belong to the specification language, but are not in the signature of the metric algebras.

Suppose (U, V) is an open exhaustion of A^u, and $f : A^u \underset{U}{\to} A_s$.

Definition 5.1.2 (*Conditional localised equational or inequality specification*)

(a) A *conditional localised Σ-equational or inequality specification of f on A w.r.t.* (U, V) has the form

$$(\Sigma', E),$$

where Σ' is an expansion of Σ by function symbols including "f" (for f), and E is a finite set of conditional Σ'-localised equations and inequalities w.r.t. $u \to s$, which specifies f uniquely on U, i.e.,

(i) $(A, f) \models E$, and

(ii) for any $f' : A^u \underset{U}{\to} A_s$, if $(A, f') \models E$ then $f' \upharpoonright U = f \upharpoonright U$.

(b) A *conditional Σ-equational or inequality specification of f on A* is defined as above, except that E contains no localising atoms.

By adapting the proof of Theorem 2 of [53, Section 6] to *local* uniform ***While****-approximability, we can show:

Theorem 5.1.3 (Universal conditional localised equational or inequality specification of ***While**** approximable functions). *For each Σ-function type $u \to s$ we can effectively find a signature $\Sigma^*_{u,s}$ which expands Σ^* by function symbols only, and a finite conditional localised equational or inequality specification*

$$\left(\Sigma^*_{u,s}, E_{u,s}(\mathrm{z})\right) \tag{8}$$

*which is universal for all locally uniformly **While***-approximable functions of type $u \to s$, w.r.t. any open exhaustion (U, V) of A^u. More precisely, the specification* (8) *contains a distinguished natural number variable* z *such that for every **While**$^*(\Sigma)$ procedure P :* $\mathsf{nat} \times u \to s$, *total metric Σ-algebra A, open exhaustion (U, V) of A and function $f : A^u \underset{U}{\to} A_s$, if f is locally uniformly approximable on A by P w.r.t.* (U, V), *then* $(\Sigma^*_{u,s}, E_{u,s}(\bar{k}))$, *where* $k = \ulcorner P \urcorner$, *specifies* f *on* A *w.r.t.* (U, V), *with hidden sorts and functions.*

Remark 5.1.4

(a) The localising atoms in the conditional specifications in this theorem are only needed on the l.h.s. of the conditional equations, and only in the form "Loc(x, n)" for variables x : u and n : nat.

(b) The "hidden sorts" mentioned are (only) the starred sorts s^* for all Σ-sorts s.

5.2. Universal specification of GL-computable functions

Now consider in particular the total real algebra $\mathscr{R}_t^N$. For convenience we display its complete definition in Fig. 2.

Let $\boldsymbol{GL}^T(\mathbb{R}^q \to \mathbb{R})$ be the class of *total* functions $f : \mathbb{R}^q \to \mathbb{R}$ that are GL-computable on $\mathbb{R}$. By combining and further developing Theorems 5.1.3 and 3.3.2 we will prove (cf. Theorem 3 of [53, Section 6], which applies to [0, 1] instead of $\mathbb{R}$):

Theorem 5.2.1 (Universal conditional equational or inequality specification of GL-computable functions). *For each* $q > 0$ *we can effectively find a signature* Σ_q *which is an expansion of* $\Sigma(\mathscr{R}_t^N)$ *by finitely many function symbols, and a finite conditional equational or inequality specification* $(\Sigma_q, E_q(\mathtt{z}))$ *which is universal for specifications of* $\boldsymbol{GL}^T(\mathbb{R}^q \to \mathbb{R})$, *in the following sense*: *it contains a distinguished natural number variable* z *such*

algebra	$\mathcal{R}_t^N$
carriers	$\mathbb{R}, \mathbb{N}, \mathbb{B},$
functions	$0_{\mathsf{real}}, 1_{\mathsf{real}} : \ \to \mathbb{R},$
	$+, \times : \mathbb{R}^2 \to \mathbb{R},$
	$- : \mathbb{R} \to \mathbb{R},$
	$\mathsf{div}_{\mathsf{nat}} : \mathbb{R} \times \mathbb{N} \to \mathbb{R},$
	$0_{\mathsf{nat}} : \ \to \mathbb{N},$
	$\mathsf{S} : \mathbb{N} \to \mathbb{N},$
	$\mathsf{tt}, \mathsf{ff} : \ \to \mathbb{B},$
	$\mathsf{and}, \mathsf{or} : \mathbb{B}^2 \to \mathbb{B},$
	$\mathsf{not} : \mathbb{B} \to \mathbb{B},$
	$\mathsf{if}_{\mathsf{real}} : \mathbb{B} \times \mathbb{R}^2 \to \mathbb{R},$
	$\mathsf{if}_{\mathsf{nat}} : \mathbb{B} \times \mathbb{N}^2 \to \mathbb{N},$
	$\mathsf{eq}_{\mathsf{nat}}, \mathsf{less}_{\mathsf{nat}} : \mathbb{N}^2 \to \mathbb{B},$
end	

Fig. 2.

that each function $f \in \boldsymbol{GL}^T(\mathbb{R}^q \to \mathbb{R})$ is specified (with hidden functions) by a suitable substitution instance $(\Sigma_q, E_q(\bar{k}))$, where k can be found effectively from a GL-code for f.

Proof. Suppose $f \in \boldsymbol{GL}^T(\mathbb{R}^q \to \mathbb{R})$. By Theorem 4.4.1, there is a $\boldsymbol{While}(\mathscr{R}_t^N)$ function

$$g : \mathbb{N} \times \mathbb{R}^q \to \mathbb{R}$$

which approximates f locally uniformly (w.r.t. the standard open exhaustion), in the sense that for all n and $x \in \mathbb{R}^q$:

$$-n < x < n \quad \Rightarrow \quad \mathsf{d}(f(x), g(n, x)) < 2^{-n}, \tag{9}$$

(cf. Remark 3.2.2) where d is the standard metric on the reals: $\mathsf{d}(x, y) = |x - y|$. Next, by the Universal Function Theorem for $\boldsymbol{While}(\mathscr{R}_t^N)$ ([52, Section 4.9], [53, Section 4.4]) there is a $\boldsymbol{While}(\mathscr{R}_t^N)$ function

$$G^q : \mathbb{N} \times \mathbb{N} \times \mathbb{R}^q \to \mathbb{R}$$

which is universal for $\boldsymbol{While}(\mathscr{R}_t^N)$ functions of type $\mathsf{nat} \times \mathsf{real}^q \to \mathsf{real}$, and so there exists k such that for all n and $x \in \mathbb{R}^q$:

$$G^q(k, n, x) = g(n, x). \tag{10}$$

Note that k can be found effectively from a $\boldsymbol{While}(\Sigma(\mathscr{R}_t^N))$ index for g, which in turn can be found effectively from a $\boldsymbol{GL}$ index for f. Note also that G^q is μPR computable on $\mathscr{R}_t^N$ [52, Theorem 8] and we can effectively find a μPR derivation for G^q from its $\boldsymbol{While}$ procedure.

From (9) and (10), for all n and $x \in A^u$,

$$-n < x < n \quad \Rightarrow \quad \mathsf{d}(f(x), G^q(k, n, x)) < 2^{-n}. \tag{11}$$

The signature Σ_q expands $\Sigma(\mathscr{R}_t^N)$ by the functions listed below, and $E_q(\mathtt{z})$ consists of conditional equational and inequality specifications for these functions, as follows:

(i) the function $\mathsf{negexp} : \mathbb{N} \to \mathbb{R}$, where $\mathsf{negexp}(n) = 2^{-n}$, used for assertions about computable approximations, together with its primitive recursive equational specification

$$\mathsf{negexp}(0_{\mathsf{nat}}) = 1_{\mathsf{real}}, \quad \mathsf{negexp}(\mathsf{S}(\mathtt{z})) = \mathsf{div}_{\mathsf{nat}}(\mathsf{negexp}(\mathtt{z}), \mathsf{S}(\mathsf{S}0));$$

(ii) the embedding $\mathsf{i}_N : \mathbb{N} \hookrightarrow \mathbb{R}$, together with its primitive recursive equational specification

$$\mathsf{i}_N(0_{\mathsf{nat}}) = 0_{\mathsf{real}}, \quad \mathsf{i}_N(\mathsf{S}(\mathtt{n})) = \mathsf{i}_N(\mathtt{n}) + 1;$$

(iii) the function f and the "universal approximating function" G^q and the conditional inequality (11) connecting the two:

$$(-\mathsf{i}_N(\mathtt{n}) < \mathtt{x}) \wedge (\mathtt{x} < \mathsf{i}_N(\mathtt{n})) \to \mathsf{d}(\mathsf{f}(\mathtt{x}), \mathsf{G}^\mathsf{q}(\mathtt{z}, \mathtt{n}, \mathtt{x})) < \mathsf{negexp}(\mathtt{n}) \tag{12}$$

which can be re-written without the metric "d" as a pair of conditional inequalities:

$$\begin{aligned} &(-\mathsf{i}_N(\mathtt{n}) < \mathtt{x}) \wedge (\mathtt{x} < \mathsf{i}_N(\mathtt{n})) \to \mathsf{f}(\mathtt{x}) < \mathsf{G}^\mathsf{q}(\mathtt{z}, \mathtt{n}, \mathtt{x}) + \mathsf{negexp}(\mathtt{n}), \\ &(-\mathsf{i}_N(\mathtt{n}) < \mathtt{x}) \wedge (\mathtt{x} < \mathsf{i}_N(\mathtt{n})) \to \mathsf{G}^\mathsf{q}(\mathtt{z}, \mathtt{n}, \mathtt{x}) < \mathsf{f}(\mathtt{x}) + \mathsf{negexp}(\mathtt{n}); \end{aligned} \tag{13}$$

(iv) the auxiliary functions used in the μPR derivation of G^q, together with equational specifications for these (cf. (v) below);

(v) the characteristic function for "bounded universal quantification" on $\mathbb{N}$, together with its primitive recursive equational specification [53, Section 3.3], which is used for an equational specification of the μ-operator [53, Section 5.2], which in turn is needed for deriving an equational specification of G^q from its μPR derivation. □

Remark 5.2.2

(a) Three simplifications are possible with regard to the signature Σ_q of this specification, compared to the more general case of Theorem 5.1.3, which applies to arbitrary total metric Σ-algebras with arbitrary open exhaustions.

(i) The localising atoms "$\mathtt{Loc}(\mathtt{x}, \mathtt{n})$" needed in the general case (cf. Remark 5.1.4(a)) can be replaced by conjunctions of inequalites

$$(-\mathtt{i}_N(\mathtt{n}) < \mathtt{x}) \wedge (\mathtt{x} < \mathtt{i}_N(\mathtt{n}))$$

as in (12) above.

(ii) The metrics d can be eliminated, as in the transformation from (12) to (13) above.

(iii) The starred sorts real^* etc. are not needed. (This follows, e.g., from the equivalence given by Theorem 3.3.2(i) and (ii).) Thus there are no hidden sorts here (cf. Remark 5.1.4(b)).

(b) This theorem is stated for total functions, and the standard exhaustion of $\mathbb{R}^q$, since those were the assumptions in Theorem 3.3.2. Again (cf. Remarks 3.3.3(c) and 4.4.2) we conjecture the theorem is true more generally.

5.3. Application: universal specification of dynamical systems

We illustrate the connection between algebraic specification methods and models of dynamical systems. (Compare [53, Section 6.3], where the following analysis was carried out for systems on $[0, 1]^q$ instead of $\mathbb{R}^q$.)

A *deterministic dynamical system* with finite dimensional state space $S \subseteq \mathbb{R}^q$ and time $T \subseteq \mathbb{R}$ is represented in a model by a function

$$\phi : T \times S \to S,$$

where for $t \in T$, $s \in S$, $\phi(t, s)$ is the state of the system at time t with initial state s. For example, the state of a particle in motion is represented by position and velocity. Thus, for a system of n particles in three-dimensional space, the state space has $6n$ dimensions.

In practice, the model is specified by ordinary differential equations (ODEs) whose complete solution is ϕ. Specifically, in the modern qualitative theory of ODEs [1], ϕ is differentiable, and the function $\phi_t : S \to S$ defined by

$$\phi_t(s) = \phi(t, s) \text{ for } t \in T, s \in S,$$

is a 1-parameter group of diffeomorphisms of S; the action of this group on S is ϕ itself, called the *flow* on the *phase space* S. This flow can be specified by a vector field on S.

In modelling a dynamical system, one aim is to *compute* values of the function ϕ on some time interval and subspace of the space of initial conditions. Many methods exist to derive algorithms for ϕ from the equations that define it. Indeed, various fields of applied mathematics exist in order to design such equations, and the field of numerical analysis exists to design such solution methods.

Conversely, we suppose that $\phi : S \to S$ can be simulated on a digital computer, i.e., ϕ is a classically computable (e.g., GL-computable) function. Note, in this connection, that in mechanics, state spaces are often modelled as differentiable manifolds of finite dimension.

By Whitney's Theorem [58] any differentiable manifold of dimension k can be embedded as a submanifold of $\mathbb{R}^{2k+1}$. So

Assume that the state space S is the Euclidean q-space $\mathbb{R}^q$, and the time dimension T is the real line $\mathbb{R}$.

Thus we have a state evolution function

$$\phi : \mathbb{R} \times \mathbb{R}^q \to \mathbb{R}^q.$$

We can now apply Theorem 5.2.1 to show that the dynamical system has a finite algebraic specification; indeed, there is one finite system of formulae that defines them all.

Theorem 5.3.1 (Universal specification of computable dynamical systems). *For each $n > 0$ there is a finite signature $\tilde{\Sigma}_n$ which extends the signature $\Sigma(\mathscr{R}_t^N)$ of the total real algebra $\mathscr{R}_t^N$ by function symbols, and a finite specification*

$$(\tilde{\Sigma}_n, \tilde{E}_n(\mathtt{z}))$$

consisting of conditional equations and inequalities, which is universal for all GL-computable dynamical systems on Euclidean n-space $\mathbb{R}^n$.

Note that $\tilde{\Sigma}_n$ is essentially the signature Σ_q of Theorem 5.2.1, with $n = q + 1$.

6. Concluding remarks and future directions

Finally, we consider a number of future directions and some open problems.

6.1. Algebraic specifications and topological algebras

Our chosen method to specify functions on a metric algebra involves:

(i) *Algebraic specifications*: We have developed formulae that are based on conditional equations, but customised for metric algebras (i.e., by introducing inequalities between reals) or the kind of function at hand (i.e., by introducing exhaustions).

We have on occasion required algebras and/or functions to be total to simplify notions of computation, approximation and/or specification. Clearly, the general case of partial multivalued functions needs further analysis.

(ii) *Unique semantics*: We have required that the formulae define the functions uniquely on a specific metric algebra, or on a class of such algebras of the same signature.

In particular, we have ignored the problem of specifying the underlying metric algebra itself; more specifically, we have ignored the problem of specifying the data of the algebra. There are a number of approaches that try to specify topological data types such as infinite tree algebras, algebras of real numbers and stream algebras, with different fields of application in mind. This brings us to

Problem 1. *To create a comprehensive algebraic theory for the specification of topological data types.*

As with models of computation, this problem can be approached in two ways.

(a) *Concrete approaches*: Some uncountable structures, such as algebras of infinite trees, were considered in the seminal work of the ADJ group [16]. Maurice Nivat and his colleagues developed these ideas of continuous algebras into an extensive theory of algebraic semantics for programming languages. Order-theoretic notions and metric methods play a vital role in this approach, as in [17]. Although the theory of data types, and the principal examples of real numbers and function spaces, are not the subject of algebraic semantics, its mathematical techniques are relevant to a theory of topological algebras. These techniques have been used to create a theory of processes, starting with [10] and extensively developed by de Bakker and co-workers [8,9].

These methods motivated the idea of representing certain topological algebras by domains [41]. Infinite trees, ordered algebras, metric and ultrametric algebras, inverse limits and domains are intimately related, and are the basis of a concrete theory of data types that deals with representations.

(b) *Abstract approaches*: The algebraic theory of data uses "algebraic formulae" to axiomatise the properties of operations and hence avoid problems of representations. Axiomatic approaches to specifying the real numbers are not new. There is David Hilbert's proof that the field of real numbers is uniquely defined as a complete archimedian ordered field, and attempts at axiomatising computer arithmetics start in [56]. A recent such specification is given in [38]. But a general specification theory for topological algebras that retains the special features of the countable case is yet to be found.

The theory of higher order algebraic specifications seems appropriate for tackling topological algebras. Higher order data types were considered by Maibaum and Lucena [28] and a theory of higher order specifications developed systematically by Bernhard Möller and colleagues, starting in [30,31], with an emphasis on refinement. Meinke studied the universal algebra of higher order specifications in a series of papers [25,26]. In addition to results on the power of higher order specifications, he considered explicitly working with (i) topological data types in general [26]; (ii) stream algebras for hardware verification [33].

Higher order algebra provides an abstract approach for a general theory of specification and verification in topological data types.

The theory of term rewriting for infinite terms [11,23] is also a mathematical tool that could be applied to this problem. A related approach, using infinite terms combined with initial model semantics, was given in [54].

We conjecture that all the above approaches work, and can be shown to be equivalent, at least for metric algebras.

6.2. Equations

Since the late 1970s, the scope and limits of specifying countable data types with equations or conditional equations, with and without hidden functions and sorts, under both initial and final semantics, has been studied in some depth; see [6] which contains a systematic account of the first phase of the programme and, for example, the recent striking results in [22].

In the topological context, several questions can be asked about the expressive power of conditional equations and inequalities.

Problem 2. *Can all computably approximable functions on a total metric algebra be defined by (conditional) equations only?*

The converse problems, exploring under what conditions algebraic specifications are computable, are largely open. For countable data types the (semi- or cosemi-)computability of specifications under initial or final semantics was easy to prove [6]. In the topological case, however, it is known that finite systems of equations can define non-computable functions [53, Section 6.4]. Now under broad conditions, a computable function must be continuous (roughly speaking), by Tseitin's Theorem. Thus we expect some form of continuity (at least) as a necessary condition to be added to equational definability to ensure computability.

Now, the property of a function f of being a homomorphism is an equational condition. It is defined by a system of equations: for each operation F in the signature, there is an equation of the form

$$f(F(\mathrm{x}_1, \ldots, \mathrm{x}_m)) = F(f(\mathrm{x}_1), \ldots, f(\mathrm{x}_m)).$$

Given the fact that homomorphisms generalise structural inductions, it is natural to ask if homomorphism equations might imply computability. Indeed, given that (some form of) continuity is a necessary condition for computability, we have the following wide ranging problem, for various classes of algebras.

Problem 3. *Suppose f is a metric algebra homomorphism from A to B. Under what extra conditions on A and f is the following true?*

$$f \textit{ is locally uniformly continuous} \iff f \textit{ is computable}.$$

The first result of this form was noted in the case of linear maps on Banach spaces by Pour-El and Richards ("First Main Theorem" in [35, p. 101]). Here *boundedness* of the linear map (which is equivalent to global uniform continuity), together with certain computability assumptions on the spaces, is shown to be equivalent to its computability. The proof involves the special properties and techniques of Banach space theory. The phenomenon has been analysed and extended to homomorphisms on partial metric algebras in [44].

6.3. Physical algorithmic models and dynamical systems

Topological data types and algebraic specifications are fundamental in many areas of computing. In scientific computation, mathematical models of dynamical systems are *specified* by sets of equations, for which algorithms are sought to compute their solutions and hence to simulate the systems. Typically, a question is:

> Given some kind of differential or integral equation that models a system, find algorithms to approximate their solution and, hence, to simulate the system.

Our Theorems 4.4.1 and 5.2.1 answer a *converse* question, namely:

> Given some algorithm that approximates the behaviour of a physical system, is there a set of algebraic formulae (e.g., equations) that defines the algorithmic model of the system?

To concentrate on finite-dimensional systems only, our results imply the following (cf. Theorem 5.3.1):

Theorem. *If a deterministic finite-dimensional physical system has a model that can be simulated to any degree of accuracy by a computer algorithm, then there exists an algebraic specification for the model. Indeed, for each n, there is a universal algebraic* *specification that captures all such computably approximable models with n-dimensional state spaces.*

Of course, the equations we produce are not the familiar differential or integral equations for modelling dynamical systems. They are more abstract equations designed to model purely algorithmic structures. Nevertheless, in our view, such results are intriguing. They certainly help delimit further the territory of a computability theory for dynamical systems. Still, we can ask:

Problem 4. *Does there exist a family of differential equations, the solutions of which include all, and only, the computably approximable finite dimensional systems?*

Given the wealth of algorithms and theory in numerical methods, it seems to us that relatively little is known about the computational and logical scope and limits of equations, which are central to the classical mathematical methods of science. Earlier [53] we have posed:

Problem 5. *Show that the theory of numerical solutions of differential and integral equations is a special instance of a general theory of algebraic specifications on metric algebras.*

A solution to this problem would play a role in the algebraic theory of co-ordinate free numerical software pioneered by Magne Haveraaen and Hans Munthe-Kaas (see [18] and other papers in that Special Issue).

The universal algebraic specifications for dynamical systems are clearly of theoretical interest. Although they are different from the classical differential and integral equations that describe most physical systems, the formulae are intimately connected with algorithms. Now, in mathematical modelling, using algorithms to model physical systems directly, without first deriving them from differential or integral equations, is a long established practice. Thus, we may pose the following:

Problem 6. *Can one develop universal algebraic specifications for classes of algorithmic models of physical systems that are physically meaningful?*

Independent algorithmic models of systems are commonplace in

(i) simulating non-linear dynamical systems, for example, in biology, where it began over 50 years ago with McCulloch's and Pitts' neural nets (in 1943) and von Neumann's cellular automata (in 1952).

They are also ubiquitous in

(ii) specifying hardware for computing devices, where algorithmic models are used to define the architecture and intended behaviour of a physical device or machine.

Computational universality has been studied in both areas. Indeed, one might also argue that the older discipline of

(iii) designing analogue computers for specific problems

could be added to the list, though differential and integral equations play a role there.

Many of the algorithms used for dynamical systems and hardware have been shown to have a common structure, namely that of *synchronous concurrent algorithms* (SCAs). SCAs are networks of processors linked by channels that are synchronised by a global clock. They occupy discrete space and operate in discrete time, processing infinite streams of data. In particular, SCAs can be defined using recursion equations over stream algebras. Thus, in SCA theory one can unify many disparate systems using a standard method for making algebraic specifications that are physically meaningful. For some basic information on SCAs see [48], for applications to dynamical systems see [19,37], and for applications to hardware see [20,21]. A study of the equational specification of general computable, countable, discrete space, discrete time dynamical and hardware systems is given in [36].

We think these are exciting and difficult problems, with many applications and ramifications.

Acknowledgements

We are grateful to two anonymous referees for helpful comments.

References

[1] V.I. Arnold, Ordinary Differential Equations, MIT Press, 1973.

[2] L. Blum, F. Cucker, M. Shub, S. Smale, Complexity and Real Computation, Springer-Verlag, 1998.

[3] V. Brattka, Recursive characterisation of computable real-valued functions and relations, Theoretical Computer Science 162 (1996) 45–77.

[4] V. Brattka, Recursive and computable operations over topological structures, Ph.D. Thesis, FernUniversität Hagen, Fachbereich Informatik, Hagen, Germany, 1999, Informatik Berichte 255, FernUniversität Hagen, July 1999.

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

[6] J.A. Bergstra, J.V. Tucker, Algebraic specifications of computable and semicomputable data types, Theoretical Computer Science 50 (1987) 137–181.

[7] J.A. Bergstra, J.V. Tucker, Equational specifications, complete term rewriting and computable and semicomputable algebras, Journal of the Association for Computing Machinery 42 (1995) 1194–1230.

[8] J.W. de Bakker, E. de Vink, Control Flow Semantics, The MIT Press, 1999.

[9] J.W. de Bakker, J.J.M.M. Rutten (Eds.), Ten Years of Concurrency Semantics, World Scientific, 1992.

[10] J.W. de Bakker, J.I. Zucker, Processes and the denotational semantics of concurrency, Information and Control 54 (1982) 70–120, Reprinted, with errata, in [9].

[11] N. Dershowitz, Rewrite, rewrite, rewrite, rewrite, rewrite, …, Theoretical Computer Science 83 (1991) 71–96.

[12] A. Edalat, Domains for computation in mathematics, physics and exact real arithmetic, Bulletin of Symbolic Logic 3 (1997) 401–452.

[13] E. Engeler, Algorithmic Properties of Structures, World Scientific Publishing Co., 1993.

[14] A. Grzegorczyk, Computable functions, Fundamenta Mathematicae 42 (1955) 168–202.

[15] A. Grzegorczyk, On the definitions of computable real continuous functions, Fundamenta Mathematicae 44 (1957) 61–71.

[16] J.A. Goguen, J.W. Thatcher, E.G. Wagner, J.B. Wright, Initial algebra semantics and continuous algebras, Journal of the Association for Computing Machinery 24 (1977) 68–95.

[17] I. Guessarian, Algebraic Semantics, Lecture Notes in Computer Science, vol. 99, Springer-Verlag, 1981.

[18] M. Haveraaen, Case study on algebraic software methodologies for scientific computing, Scientific Programming 8 (2000) 261–273.

[19] A.V. Holden, M. Poole, J.V. Tucker, H. Zhang, Coupled map lattices as computational systems, Chaos 2 (1992) 367–376.

[20] N.A. Harman, J.V. Tucker, Algebraic models of microprocessors: architecture and organisation, Acta Informatica 33 (1996) 421–456.

[21] K.M. Hobley, B.C. Thompson, J.V. Tucker, Specification and verification of synchronous concurrent algorithms: a case study of a convoluted algorithm, in: G. Milne (Ed.), The Fusion of Hardware Design and

Verification (Proceedings of IFIP Working Group 10.2 Working Conference), North Holland, 1988, pp. 347–374.
[22] B.M. Khoussainov, Randomness, computability, and algebraic specifications, Annals of Pure & Applied Logic 91 (1) (1998) 1–15.
[23] J.R. Kennaway, J.W. Klop, M.R. Sleep, F.J. de Vries, Transfinite reductions in orthogonal term rewriting systems, Information and Computation 119 (1995) 18–38.
[24] D. Lacombe, Extension de la notion de fonction récursive aux fonctions d'une ou plusieurs variables réelles, I, II, III, Comptes Rendus De L Academie Des Sciences, Paris 240 (1955) 2470–2480, 241:13–14, 151–153.
[25] K. Meinke, Universal algebra in higher types, Theoretical Computer Science 100 (1992) 385–417.
[26] K. Meinke, Topological methods for algebraic specification, Theoretical Computer Science 166 (1996) 263–290.
[27] J. Meseguer, J.A. Goguen, Initiality, induction and computability, in: M. Nivat, J. Reynolds (Eds.), Algebraic Methods in Semantics, Cambridge University Press, 1985, pp. 459–541.
[28] T.S.E. Maibaum, C.J. Lucena, Higher-order data types, International Journal of Computing and Information Science 9 (1980) 31–53.
[29] J. Meseguer, L. Moss, J.A. Goguen, Final algebra, cosemicomputable algebras, and degrees of unsolvability, Theoretical Computer Science 100 (1992) 267–302.
[30] B. Möller, On the algebraic specification of infinite objects—ordered and continuous models of algebraic types, Acta Informatica 22 (1985) 537–578.
[31] B. Möller, Higher-order algebraic specifications, Habilitationsschrift, Technische Universität München, 1987.
[32] Y.N. Moschovakis, Recursive metric spaces, Fundamenta Mathematicae 55 (1964) 215–238.
[33] K. Meinke, J. Steggles, Specification and verification in higher order algebra: a case study of convolution, in: J. Heering, K. Meinke, B. Möller, T. Nipkow (Eds.), Higher Order Algebra, Logic and Term Rewriting, Lecture Notes in Computer Science, vol. 816, Springer-Verlag, 1994, pp. 189–222.
[34] M.B. Pour-El, J.C. Caldwell, On a simple definition of computable function of a real variable with applications to functions of a complex variable, Zeitschrift für mathematische Logik und Grundlagen der Mathematik 21 (1975) 1–19.
[35] M.B. Pour-El, J.I. Richards, Computability in Analysis and Physics, Springer-Verlag, 1989.
[36] M.J. Poole, J.V. Tucker, A.V. Holden, Hierarchies of spatially extended systems and synchronous concurrent algorithms, in: B. Möller, J.V. Tucker (Eds.), Prospects for Hardware Foundations, Lecture Notes in Computer Science, vol. 1546, Springer-Verlag, 1998, pp. 184–235.
[37] M.J. Poole, J.V. Tucker, A.V. Holden, Hierarchical reconstructions of cardiac tissue, Chaos, Solitons and Fractals 13 (2002) 1581–1612.
[38] M. Roggenbach, L. Schröder, T. Mossakowski, Specifying real numbers in CASL, in: C. Choppy, D. Bert, P. Mosses (Eds.), Recent Developments in Algebraic Development Techniques. 14th International Workshop, WADT'99, Chateau de Bonas, France, Lecture Notes in Computer Science, vol. 1827, Springer-Verlag, 2000, pp. 146–161.
[39] W. Rudin, Principles of Mathematical Analysis, third ed., McGraw-Hill, 1976.
[40] J.C. Shepherdson, On the definition of computable function of a real variable, Zeitschrift für mathematische Logik und Grundlagen der Mathematik 22 (1976) 391–402.
[41] V. Stoltenberg-Hansen, J.V. Tucker, Complete local rings as domains, Journal of Symbolic Logic 53 (1988) 603–624.
[42] V. Stoltenberg-Hansen, J.V. Tucker, Effective algebras, in: S. Abramsky, D. Gabbay, T. Maibaum (Eds.), Handbook of Logic in Computer Science, vol. 4, Oxford University Press, 1995, pp. 357–526.
[43] V. Stoltenberg-Hansen, J.V. Tucker, Concrete models of computation for topological algebras, Theoretical Computer Science 219 (1999) 347–378.
[44] V. Stoltenberg-Hansen, J.V. Tucker, Computable and continuous homomorphisms on metric partial algebras, Bulletin of Symbolic Logic 9 (2003) 299–334.
[45] D. Spreen, On effective topological spaces, Journal of Symbolic Logic 63 (1998) 185–221.
[46] D. Spreen, Representations versus numberings: on the relationship of two computability notions, Theoretical Computer Science 263 (2001) 473–499.
[47] K. Stewart, Concrete and abstract models of computation over metric algebras, Ph.D. Thesis, Department of Computer Science, University of Wales, Swansea, 1998.
[48] B.C. Thompson, J.V. Tucker, Algebraic specification of synchronous concurrent algorithms and architectures, Research Report CSR 9-91, Department of Computer Science, University College of Swansea, 1994 (second ed.).

[49] J.V. Tucker, J.I. Zucker, Program Correctness over Abstract Data Types, with Error-State Semantics, CWI Monographs, vol. 6, North Holland, 1988.
[50] J.V. Tucker, J.I. Zucker, Examples of semicomputable sets of real and complex numbers, in: J.P. Myers Jr., M.J. O'Donnell (Eds.), Constructivity in Computer Science: Summer Symposium, San Antonio, Texas, June 1991, Lecture Notes in Computer Science, vol. 613, Springer-Verlag, 1992, pp. 179–198.
[51] J.V. Tucker, J.I. Zucker, Computation by 'while' programs on topological partial algebras, Theoretical Computer Science 219 (1999) 379–420.
[52] J.V. Tucker, J.I. Zucker, Computable functions and semicomputable sets on many-sorted algebras, in: S. Abramsky, D. Gabbay, T. Maibaum (Eds.), Handbook of Logic in Computer Science, vol. 5, Oxford University Press, 2000, pp. 317–523.
[53] J.V. Tucker, J.I. Zucker, Abstract computability and algebraic specification, ACM Transactions on Computational Logic 3 (2002) 279–333.
[54] J.V. Tucker, J.I. Zucker, Infinitary initial algebra specifications for stream algebras, in: W. Sieg, R. Sommer, C. Talcott (Eds.), Reflections on the Foundations of Mathematics: Essays in Honor of Solomon Feferman, Lecture Notes in Logic, vol. 15, Association for Symbolic Logic, 2002, pp. 234–256.
[55] J.V. Tucker, J.I. Zucker, Abstract versus concrete computation on metric partial algebras, ACM Transactions on Computational Logic, in press.
[56] A. van Wijngaarden, Numerical analysis as an independent science, BIT 6 (1966) 68–81.
[57] K. Weihrauch, Computable Analysis: An Introduction, Springer-Verlag, 2000.
[58] H. Whitney, Differentiable manifolds, Annals of Mathematics 37 (1935) 645–680.