Accepted Manuscript

A topological view on algebraic computation models

Eike Neumann, Arno Pauly

PII: S0885-064X(17)30076-6
DOI: http://dx.doi.org/10.1016/j.jco.2017.08.003
Reference: YJCOM 1338

To appear in: *Journal of Complexity*

Received date : 4 March 2016
Accepted date : 2 August 2017

Please cite this article as: E. Neumann, A. Pauly, A topological view on algebraic computation models, *Journal of Complexity* (2017), http://dx.doi.org/10.1016/j.jco.2017.08.003

A topological view on algebraic computation models

Eike Neumann[a], Arno Pauly[b,c,1]

[a] *Aston University, Birmingham, UK*
[b] *University of Cambridge, Cambridge, UK*
[c] *Birmingham University, Birmingham, UK*

Abstract

We investigate the topological aspects of some algebraic computation models, in particular the BSS-model. Our results can be seen as bounds on how different BSS-computability and computability in the sense of computable analysis can be. The framework for this is Weihrauch reducibility. As a consequence of our characterizations, we establish that the solvability complexity index is (mostly) independent of the computational model, and that there thus is common ground in the study of non-computability between the BSS and TTE setting.

Keywords: Weihrauch reducibility, BSS-machine, Analytic machine, Effective DST, solvability complexity index, TTE, Computable analysis
2010 MSC: 03D78, 68Q05, 12Y05

1. Introduction

There are two major paradigms for computability on functions on the real numbers: On the one hand, computable analysis in the tradition of Grzegorczyk [1, 2] and Lacombe [3] as championed by Weihrauch [4, 5] (see also the equivalent approaches by Pour-El and Richards [6] or Ko [7]). On the other hand, the BSS-machines by Blum, Shub and Smale [8, 9], or the very similar real-RAM model. Incidentally, both schools claim to be in the tradition of Turing.

Computable analysis can, to a large extent, be understood as effective topology [10, 11] – this becomes particularly clear when one moves beyond just the real numbers, and is interested in computability on spaces of subsets or functionals. In particular, we find that the effective Borel hierarchy occupies the position analogous to the arithmetical hierarchy in classical recursion theory; and that incomputability of natural problems is typically a consequence of dis-

Email addresses: neumaef1@aston.ac.uk (Eike Neumann), Arno.M.Pauly@gmail.com (Arno Pauly)
[1] Pauly has since moved to the Université Libre de Bruxelles.

continuity. A more fine-grained view becomes possible in the framework of Weihrauch reducibility (more below).

In contrast, the study of BSS-computability is essentially a question akin to (logical) definability in algebraic structures. This causes the lack of a stable notion of BSS-computability on *the reals*: BSS-computability on the ring $(\mathbb{R}, +, \times, =)$ differs from BSS-computability on the unordered field $(\mathbb{R}, +, \times, -, /, =)$, which in turn differs from BSS-computability on the ordered field $(\mathbb{R}, +, \times, -, /, <)$. Taking into account as basic functions further maps such as square root or the exponential function induces additional variants. There are, however, also topological obstacles to being BSS-computable – and as we shall demonstrate, these obstacles are common to all variants of BSS-machines.

Hotz and his coauthors [12, 13, 14] have introduced and studied a number of extensions of BSS-machines (called *analytic machines*), which can (in various ways) make use of approximations. These additional features enable the "computation" of even more discontinuous functions, however, there are still topological limitations.

In the present article, we characterize the maximal degrees of discontinuity for functions computable by each of the algebraic machine models in the framework of Weihrauch reducibility. We thus continue and extend the work by Gärtner and Ziegler in [15]. This allows us to differentiate between topological and algebraic reasons for non-computability in these models. Moreover, we can show that for sufficiently discontinuous problems the difference between the various computational models vanishes: This can be formalized using Hansen's *solvability complexity index* (SCI) [16, 17], which is the least number of limits one needs in order to solve a particular problem over some basic computational model. Our results show e.g. that for values of 2 or greater the SCI based on the BSS-model coincides with the SCI based on the computable analysis model.

In Section 3 we characterise the computational power of generalized register machines with LPO-tests (Proposition 10) as well as BSS-machines over the reals with equality- and order-tests (Corollary 14). We show that their strength is captured by the Weihrauch degree $\mathrm{C}_{\mathbb{N}}$ of closed choice over the natural numbers, in the sense that any function computable by either type of machine is Weihrauch-reducible to $\mathrm{C}_{\mathbb{N}}$ and for each type of machine there exists a function which is Weihrauch-equivalent to $\mathrm{C}_{\mathbb{N}}$ and computable by that type of machine. We furthermore show that all BSS-computable *total* functions are strictly below $\mathrm{C}_{\mathbb{N}}$ (Corollary 18 and Proposition 19) and that the strength of BSS-machines with equality-tests but without order-tests is characterised by the strictly weaker Weihrauch degree LPO^* (Proposition 20).

In Section 4 we study the BSS-halting problem from the point of view of Weihrauch-reducibility. We show in particular that the halting problem of a machine over the signature $(\mathbb{R}, +, =, <)$ cannot be solved by any machine over a signature which expands $(\mathbb{R}, +, =, <)$ by continuous operations only. Therefore, the inability of BSS-machines to solve their halting problem already holds for topological reasons.

In Section 5 we prepare the discussion of analytic machines and the solvability complexity index by some more technical results on Weihrauch degrees.

We introduce the Weihrauch degree of the problem Sort of "sorting" an infinite binary sequence. We discuss its position in the Weihrauch lattice in detail and prove in particular that $\mathsf{Sort}^n <_W \mathsf{Sort}^{n+1}$ (Corollary 27) and that we have the absorption $\lim\star\lim\star\mathsf{Sort} \equiv_W \lim\star\lim$ (Corollary 32).

In Section 6 we characterise the strength of analytic machines. We show that the computational power of analytic machines is characterised by the Weihrauch degree Sort^* in the same way as the power of BSS-machines is characterised by $\mathrm{C}_{\mathbb{N}}$ (Observation 35 and Corollary 39).

In Section 7 we combine the results of the two previous sections to show that the SCI of an uncomputable function over the BSS-model is the same as the SCI over the computable analysis model as soon as the SCI over either model is greater than or equal to 2 (Theorem 44).

2. Background on the models and Weihrauch reducibility

2.1. Algebraic computation models

We introduce the notion of a register machine over some algebraic structure, following Gassner [18, 19, 20] (1997+) and Tavana and Weihrauch (2011) [21]. Other approaches to computation over algebraic structures were put forth e.g. by Tucker and Zucker (2000) [22] and Hemmerling (1998) [23]. For this consider some algebraic structure $\mathfrak{A} = (A, f_1, f_2, \ldots, T_1, T_2, \ldots)$, where A is a set, each f_i is a (partial) function of type $f_i :\subseteq A^{k_i} \to A$, and each T_i is a relation of type $T_i \subseteq A^{l_i}$. In the usual examples, the signatures will be finite, but this is not essential for our considerations.

Generalized register machines will compute functions of type $g : A^* \to A^*$. They have registers $(R_i)_{i\in\mathbb{N}}$ holding elements of A, and index registers $(I_n)_{n\in\mathbb{N}}$ holding natural numbers. Programs are finite lists of commands, consisting of:

- standard register machine operations on the index registers
- copying the value of the register R_{I_1} indexed by I_1 into R_{I_0}
- applying some f_i to the values contained in $R_1, \ldots, R_{k_i}$ and writing the result into R_0
- branching to a line in the program depending on the value of T_i on the values contained in $R_1, \ldots, R_{l_i}$
- HALT, in which case the values currently in the registers $R_0, \ldots, R_{I_0}$ constitute the output

Initially, the register I_0 contains the length of the input, all other I_n start at 0. The input is in $R_1, \ldots, R_n$, all other R_i contain some fixed value $a_0 \in A$. If the program either fails to halt on some input, or invokes a partial function on some values outside its domain, the computed function is undefined on these values. We call a (partial) function $g :\subseteq A^* \to A^*$ $\mathfrak{A}$-computable, if there is a generalized register machine program computing it.

In analogy to the situation in computable analysis, we shall call sections of $\mathfrak{A}$-computable function $\mathfrak{A}$-continuous. This boils down to programs being allowed additional assignment operations $R_i := a$ for any element $a \in A$. These are usually included in the algebraic models, but have the undesirable consequence of there being more than countably many programs, if A itself is uncountable.

The primary example of a structure will be $(\mathbb{R}, +, \times, <)$ (yielding BSS-computability). Secondary examples include $(\mathbb{R}, +, =)$ (additive machines with equality) and other combinations of continuous functions and tests in $\{=, <\}$.

2.2. Analytic machines

The analytic machines introduced in [12, 13] enhance BSS-machines by means to approximate functions. More generally, if we consider the algebraic structure $\mathfrak{A}$ to also carry a metric, we can define functions computable by strongly $\mathfrak{A}$-analytic machines and functions computable by weakly $\mathfrak{A}$-analytic machines in one of two equivalent ways: Either a generalized register machine receives an additional input $n \in \mathbb{N}$; thus computes a function $G :\subseteq \mathbb{N} \times A^* \to A^*$, which is then considered as weakly approximating $g :\subseteq A^* \to A^*$ iff $\forall \mathbf{a} \in \mathrm{dom}(g)\ \lim_{n\to\infty} G(n, \mathbf{a}) = g(a)$, and as strongly approximating g iff $\forall \mathbf{a} \in \mathrm{dom}(g)\ \forall n \in \mathbb{N}\ d(G(n, \mathbf{a}), g(a)) < 2^{-n}$. Alternatively there is no extra input, and the machine keeps running, and produces an infinite sequence $p \in A^\omega$ (plus some information on the length of the desired output). The limit conditions are the same.

We shall speak just of "functions computable by a strongly (weakly) analytic machine" if the underlying structure is $(\mathbb{R}, +, \times, <)$. As functions computable by an analytic machine receive their input exactly, but produce their output in an approximative fashion, they are generally not closed under composition.

Further variants of analytic machines have been considered, which are not relevant for the present paper though. We refer to [24] by Ziegler for an excellent discussion.

2.3. Type-2 Turing machines

The fundamental model for computable analysis/the TTE-framework [5] are the Type-2 Turing machines. Structurally, these do not differ from ordinary Turing machines with a designated write-once only output tape. What differs is that instead of halting and thus producing a finite output, the machine continues to run for ever. As long as it keeps writing on the output tape, this produces an infinite sequence in the limit. While some might object to the use of infinitely long computations, this model is realistic in as far as anything written on the output tape at some finite time constitutes a prefix to the infinite output (as each cell in the output tape can be changed just once). Thus, this model inherently captures approximating computations – and its intricate connection to topology is perhaps unsurprising. In particular, we find that any computable function is automatically continuous w.r.t. the standard topology on $\{0,1\}^{\mathbb{N}}$ – and vice versa, every continuous function becomes computable relative to some oracle.

Type-2 machines natively provide us with a notion of computability on $\{0,1\}^{\mathbb{N}}$. This is then transferred to the spaces of actual interest by means

of a representation. Our presentation follows [11], to which we refer for more details. A *represented space* is a pair $\mathbf{X} = (X, \delta_X)$ of a set X and a partial surjection $\delta_X :\subseteq \{0,1\}^{\mathbb{N}} \to X$. A (multi-valued) function between represented spaces is a partial relation $f \subseteq X \times Y$. We write $f :\subseteq \mathbf{X} \rightrightarrows \mathbf{Y}$ for these; here $\subseteq$ denotes (potential) partiality, and $\rightrightarrows$ (potential) multi-valuedness. We write $f(x) = \{y \in Y \mid (x,y) \in f\}$ and $\mathrm{dom}(f) := \{x \in \mathbf{X} \mid \exists y \in f(x)\}$. We recall that composition of multi-valued functions is defined via $\mathrm{dom}(g \circ f) := \{x \in \mathrm{dom}(f) \mid f(x) \subseteq \mathrm{dom}(g)\}$ and $z \in (g \circ f)(x)$ for $x \in \mathrm{dom}(g \circ f)$, if $\exists y \in f(x) \;\; z \in g(y)$.

For $f :\subseteq \mathbf{X} \rightrightarrows \mathbf{Y}$ and $F :\subseteq \{0,1\}^{\mathbb{N}} \to \{0,1\}^{\mathbb{N}}$, we call F a realizer of f (notation $F \vdash f$), iff $\delta_Y(F(p)) \in f(\delta_X(p))$ for all $p \in \mathrm{dom}(f\delta_X)$. A function between represented spaces is called computable (continuous), iff it has a computable (continuous) realizer. Note that, unlike in the case of algebraic models, the behaviour of a Type-2 machine which computes a partial function is completely unconstrained on inputs outside of the function's domain.

$$\begin{array}{ccc} \{0,1\}^{\mathbb{N}} & \xrightarrow{F} & \{0,1\}^{\mathbb{N}} \\ \downarrow{\scriptstyle \delta_{\mathbf{X}}} & & \downarrow{\scriptstyle \delta_{\mathbf{Y}}} \\ \mathbf{X} & \xrightarrow{f} & \mathbf{Y} \end{array}$$

Figure 1: The notion of a realizer

As we are primarily interested in computability on $\mathbb{R}$ and $\mathbb{R}^*$, we shall introduce the standard representations for these spaces. Fix some standard enumeration $\nu_{\mathbb{Q}} : \mathbb{N} \to \mathbb{Q}$. Then let $\rho(p) = x$ if $p = 0^{n_0}10^{n_1}1\ldots$ and $\forall i \in \mathbb{N} \;\; d(x, \nu_{\mathbb{Q}}(n_i)) < 2^{-i}$. In other words, a name for a real number is a sequence of rationals converging to it with some prescribed speed. We thus understand $\mathbb{R}$ to be the represented space $(\mathbb{R}, \rho)$. As we can form products and coproducts of represented spaces, we automatically obtain a representation for $\mathbb{R}^* = \coprod_{n\in\mathbb{N}} \mathbb{R}^n$.

We will encounter decision problems, and thus need spaces of truth-values. For this, we use both the space $\{0,1\}$ represented by $\delta_{\mathbf{2}}$ defined by $\delta_{\mathbf{2}}(p) = p(0)$, as well as Sierpiński space $\mathbb{S}$. The latter has the underlying set $\{\bot, \top\}$ and the representation $\delta_{\mathbb{S}}$ with $\delta_{\mathbb{S}}(0^\omega) = \bot$ and $\delta_{\mathbb{S}}(p) = \top$ iff $p \neq 0^\omega$. As usual, we identify 0 with $\bot$ and 1 with $\top$. The space $\{0,1\}$ captures decidability, and $\mathbb{S}$ captures semi-decidability. The usual boolean connectives $\wedge$ and $\vee$ are computable on both spaces. Negation $\neg : \{0,1\} \to \{0,1\}$ is computable, whereas $\neg : \mathbb{S} \to \mathbb{S}$ is not computable.

We also make use of the represented space $\mathbb{N}$, represented via $\delta_{\mathbb{N}}^{-1}(n) = \{0^n1^\omega\}$. Any represented space naturally comes with a topology, namely the final topology along the representation, where the domain of the representation just inherits the subspace topology of the usual complete metric on $\{0,1\}^{\mathbb{N}}$. For $\mathbb{N}$, this is the discrete topology, for $\mathbb{R}$ the usual Euclidean topology. Via the utm-theorem, we obtain the represented space $\mathcal{O}(\mathbf{X})$ of the open subsets of the represented space $\mathbf{X}$ in a canonical manner, by identifying them with continuous functions from X to $\mathbb{S}$. By considering complements instead we

obtain a representation of the space $\mathcal{A}(\mathbf{X})$. Here, the space $\mathcal{A}(\mathbf{X})$ is the space of closed subsets represented with *negative* information, or equipped with the upper Fell topology. A representation via *positive* information is obtained by identifying a closed set $A \subseteq \mathbf{X}$ with the open set of all open sets which intersect A. The corresponding represented space is denoted by $\mathcal{V}(\mathbf{X})$ and called the space of *overt subsets* of $\mathbf{X}$. We only use the special cases $\mathcal{O}(\mathbb{N})$ and $\mathcal{A}(\mathbb{N})$ here though. One may consider $\mathcal{O}(\mathbb{N})$ to be represented by $\delta\colon \{0,1\}^{\mathbb{N}} \to \mathcal{O}(\mathbb{N})$ with $n \in \delta(p)$ iff $01^{n+1}0$ is a subword of p; and thus $\mathcal{A}(\mathbb{N})$ by $\psi\colon \{0,1\}^{\mathbb{N}} \to \mathcal{A}(\mathbb{N})$ with $n \in \psi(p)$ iff $01^{n+1}0$ is not a subword of p. Note that the computable points of $\mathcal{O}(\mathbb{N})$ are precisely the computably enumerable sets, and the computable points of $\mathcal{A}(\mathbb{N})$ are the co-c.e. sets.

2.4. Weihrauch reducibility

Weihrauch reducibility is a preorder on multivalued functions between represented spaces, and serves as a framework for comparing incomputability in the Type-2 setting, similar to the role of many-one or Turing reductions in classical recursion theory. We refer to [25] for a more detailed introduction, and a survey of known results.

Definition 1 (Weihrauch reducibility)**.** Let f, g be multi-valued functions on represented spaces. Then f is said to be *Weihrauch reducible* to g, in symbols $f \leq_{\mathrm{W}} g$, if there are computable functions $K, H :\subseteq \{0,1\}^{\mathbb{N}} \to \{0,1\}^{\mathbb{N}}$ such that $K\langle \mathrm{id}, GH\rangle \vdash f$ for all $G \vdash g$. Accordingly, f is said to be *continuously Weihrauch reducible* to g, in symbols $f \leq^{c}_{\mathrm{W}} g$, if there exist continuous functions K and H satisfying this condition.

The relation $\leq_{\mathrm{W}}$ is reflexive and transitive. We use $\equiv_{\mathrm{W}}$ to denote equivalence regarding $\leq_{\mathrm{W}}$, and by $<_{\mathrm{W}}$ we denote strict reducibility. By $\mathfrak{W}$ we refer to the partially ordered set of equivalence classes. As shown in [26, 27], $\mathfrak{W}$ is a distributive lattice, and also the usual product operation on multivalued function induces an operation $\times$ on $\mathfrak{W}$. The algebraic structure on $\mathfrak{W}$ has been investigated in further detail in [28, 29].

There are two relevant unary operations defined on $\mathfrak{W}$, both happen to be closure operators. The operation * was introduced in [26, 30] by setting $f^0 := \mathrm{id}_{\mathbb{N}^{\mathbb{N}}}$, $f^{n+1} := f \times f^n$ and then $f^*(n, x) := f^n(x)$. It corresponds to making any finite number of parallel uses of f available. Similarly, the *parallelization* operation $\widehat{\ }$ from [27, 31] makes countably many parallel uses available by $\widehat{f}(x_0, x_1, x_2, \ldots) := (f(x_0), f(x_1), f(x_2), \ldots)$.

We will make use of an operation $\star$ defined on $\mathfrak{W}$ that captures aspects of function composition. Following [32, 33], let $f \star g := \max_{\leq_{\mathrm{W}}}\{f_0 \circ g_0 \mid f \equiv_{\mathrm{W}} f_0 \wedge g \equiv_{\mathrm{W}} g_0\}$. We understand that the quantification is running over all suitable functions f_0, g_0 with matching types for the function composition. It is not obvious that this maximum always exists, this is shown in [29] using an explicit construction for $f \star g$. Like function composition, $\star$ is associative but generally not commutative. We use $\star$ to introduce iterated composition by setting $f^{(0)} := \mathrm{id}_{\mathbb{N}^{\mathbb{N}}}$ and $f^{(n+1)} = f^{(n)} \star f$.

All computable multivalued functions with a computable point in their domain are Weihrauch equivalent, this degree is denoted by 1.

An important source for examples of Weihrauch degrees that are relevant for the classification of theorems are the closed choice principles studied in e.g. [31, 34]:

Definition 2. Given a represented space $\mathbf{X}$, the associated closed choice principle $\mathrm{C}_{\mathbf{X}}$ is the partial multivalued function $\mathrm{C}_{\mathbf{X}} :\subseteq \mathcal{A}(\mathbf{X}) \rightrightarrows \mathbf{X}$ mapping a non-empty closed set to an arbitrary point in it.

The Weihrauch degree corresponding to $\mathrm{C}_{\mathbb{N}}$ has received significant attention, e.g. in [31, 34, 35, 36, 37, 38, 39, 40, 41]. In particular, as shown in [42], a function between computable Polish spaces is Weihrauch reducible to $\mathrm{C}_{\mathbb{N}}$ iff it is piecewise computable iff it is effectively Δ^0_2-measurable.

The second standard Weihrauch degree very relevant for our investigation will be lim, with its representative $\lim :\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ defined via $\lim(p)(n) = \lim_{i\to\infty} p(\langle n, i\rangle)$. It was shown in [43] that lim is Weihrauch-complete for Σ^0_2-measurable functions, and that, more generally, $\lim^{(n)}$ is Weihrauch-complete for Σ^0_{n+1}-measurable function. This line of research was continued in [44].

The third standard Weihrauch degree we will refer to is LPO, which has the eponymous representative $\mathrm{LPO} : \{0,1\}^{\mathbb{N}} \to \{0,1\}$ mapping 0^ω to 1 and $p \neq 0^\omega$ to 0. We also find $\mathrm{id} : \mathbb{S} \to \{0,1\}$ in this class. Furthermore, $= 0 : \mathbb{R} \to \{0,1\}$, $= : \mathbb{R}\times\mathbb{R} \to \{0,1\}$ and $< : \mathbb{R}\times\mathbb{R} \to \{0,1\}$ are members of the Weihrauch degree LPO. In the context of computable analysis the degree was first introduced and named as such in [45], based on earlier usage in constructive mathematics [46].

For our purposes, the following representatives and properties of the degree $\mathrm{C}_{\mathbb{N}}$ are also relevant:

Lemma 3 ([40])**.** The following are Weihrauch equivalent:

1. $\mathrm{C}_{\mathbb{N}}$, that is closed choice on the natural numbers.
2. $\mathrm{UC}_{\mathbb{N}}$, defined via $\mathrm{UC}_{\mathbb{N}} = (\mathrm{C}_{\mathbb{N}})\,|_{\{A\in\mathcal{A}(\mathbb{N})||A|=1\}}$.
3. $\min :\subseteq \mathcal{A}(\mathbb{N}) \to \mathbb{N}$ defined on the non-empty closed subsets of $\mathbb{N}$.
4. $\max_{\mathcal{O}} :\subseteq \mathcal{O}(\mathbb{N}) \to \mathbb{N}$ defined on the non-empty bounded open subsets of $\mathbb{N}$.
5. $\max_{\mathbb{N}^{\mathbb{N}}} :\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$ defined by $\max_{\mathbb{N}^{\mathbb{N}}}(p) = \max\{p(i) \mid i \in \mathbb{N}\}$.
6. $\mathrm{Bound} :\subseteq \mathcal{O}(\mathbb{N}) \rightrightarrows \mathbb{N}$, where $n \in \mathrm{Bound}(U)$ iff $\forall m \in U\ n \geq m$.

Lemma 4 ([32])**.** The following are Weihrauch equivalent:

1. $\mathrm{C}_{\mathbb{N}}$
2. $\lim_\Delta :\subseteq \mathbb{R}^{\mathbb{N}} \to \mathbb{R}$, where $\lim_\Delta$ maps an eventually constant sequence to its limit

Lemma 5 ([36, 35])**.** 1. For $f : \mathbf{X} \to \{0,\dots,n\}$ we have $\mathrm{C}_{\mathbb{N}} \not\leq_{\mathrm{W}} f$.
2. $\mathrm{LPO} <_{\mathrm{W}} \mathrm{LPO}^* <_{\mathrm{W}} \mathrm{C}_{\mathbb{N}}$

Lemma 6 ([34])**.** 1. $\mathrm{C}_{\mathbb{N}} \star \mathrm{C}_{\mathbb{N}} \equiv_{\mathrm{W}} \mathrm{C}_{\mathbb{N}}$
2. $\lim \star \mathrm{C}_{\mathbb{N}} \equiv_{\mathrm{W}} \lim$

2.5. On the difference between algebraic and topological models of computation

Penrose [47] posed and made popular the question whether the Mandelbrot set is computable – albeit without specifying any formal definition of *computable* for this question. In the BSS-model, a negative answer was readily obtained in [9]. Brattka [48], however, argued that this result just reflects that the Mandelbrot set is not an algebraic object, without being meaningful for computability as naively understood: A very similar proof applies also to the epigraph of the exponential function – which, as many[2] would agree, *ought* to be computable.([3])

The usual focal point for disagreement between the two communities, however, is not about functions computable in the Type-2 sense but non-computable in the BSS-sense, but vice versa. As is commonly understood, and will be proven formally below, this boils down to equality (or any other non-trivial property of real numbers) being decidable in the BSS-model. Brattka and Hertling [51] proposed the *feasible real RAM*'s, a variation on the BSS-model that rather than deterministic tests can only perform non-deterministic tests that may give a wrong answer for very close numbers. The functions approximable by a feasible real RAM are precisely those computable in the Type-2 sense; i.e. allowing approximation and removing exact tests is precisely what is needed to move from BSS-computability to Type-2 computability.

The complexity of (semi)decidable sets in the two models was compared by Zhong [52], and Boldi and Vigna [53]. In [52] it is proved that every TTE-semi-decidable set is BSS semi-decidable, and a criterion is given under which the converse holds true. In [53] it is shown amongst other things that if an open set $U \in \mathcal{O}(\mathbb{R}^n)$ is BSS decidable with constants $c_1, \ldots, c_n$, then the Turing degree of some standard name of U is below the jump of the degrees of the binary expansion of the constants. Furthermore, a particular set U is constructed for which the degree of every standard name of U is above the jump of the degrees of the binary expansion of the constants. It is also shown that the halting set of any BSS machine with constants $c_1, \ldots, c_n$ is computably (Turing-)overt relative to the constants.

We can translate some of their results into our parlance. Let OD denote the space of open subsets of $\mathbb{R}^*$ that are decidable by a BSS-machine using constants, represented in the obvious way by a Gödel-number of the machine together with names for the constants. Let SD denote the space of Halting sets of BSS-machines using constants, again represented in the obvious way. Then:

Proposition 7. 1. $\mathrm{id} : \mathcal{O}(\mathbb{R}^*) \to \mathrm{SD}$ is computable.
2. $\lim \leq_{\mathrm{W}} (\mathrm{id} : \mathrm{OD} \to \mathcal{O}(\mathbb{R}^*))$.
3. There is some $g :\subseteq \mathbb{N}^{\mathbb{N}} \rightrightarrows \mathbb{N}$ such that $(\mathrm{id} : \mathrm{OD} \to \mathcal{O}(\mathbb{R}^*)) \leq_{\mathrm{W}} g \star \lim$.

[2] Including Penrose himself, see e.g. [47, Figure 4.5, p. 167].

[3] The question whether the (distance function of the) Mandelbrot set is computable in the computable analysis sense is still open – as shown by Hertling [49], this would be implied by the hyperbolicity conjecture. See the book [50] for a general discussion of Julia sets and computability.

4. There is some $g :\subseteq \mathbb{N}^{\mathbb{N}} \rightrightarrows \mathbb{N}$ such that $\left(\overline{\ } : \mathrm{SD} \to \mathcal{V}(\mathbb{R}^*)\right) \leq_{\mathrm{W}} g$, where $\overline{\ }$ denotes the closure operator.

Proof. 1. This is the uniform version of [52, Theorem 3.1].
2. This follows by noting that the proof of [53, Theorem 10] is completely uniform.
3. This follows from [53, Theorem 7].
4. This is the statement of [53, Corollary 6].

□

While making these classifications precise, and attending to the numerous remaining questions on the complexity of sets in terms of Weihrauch reducibility seems like an interesting endeavour, we leave it to future work.

In [54], the notion of semi-decidability in the TTE model is related to Σ-definability over the reals without equality: It is shown that the Σ-definable sets without equality are precisely the semi-decidable sets in the TTE-model. The main result of the paper is that there is no effective procedure which takes as input a finite formula that defines an open set with equality and takes it to a formula that defines the same open set without equality.

2.6. Relativization

Prima facie, several of our most general results might look unsatisfactory to the reader coming from the algebraic computation model side: We restrict our operations to be computable, hence presuppose the Type-2 notion of computability, and disallow the use of arbitrary constants that is customary for BSS-machines. These issues can be resolved directly, using the technique of *relativization* from classical recursion theory. The idea here is that almost all computability-theoretic arguments remain true relative to some arbitrary, but fixed oracle $\Omega \in \{0,1\}^{\mathbb{N}}$. This is of crucial importance for us due to the following:

Fact 8. A function is continuous iff it is computable relative to some oracle.

All of our proofs relativize. Thus, for each of our results replacing each instance of *computable* by *continuous* again yields a true statement. Moreover, for the relativized version of a statement about an algebraic computation model, it does not change anything to allow arbitrary constants.

3. The complexity of finitely many tests

3.1. Generalized register machines with LPO-tests

As explained above, the crucial distinguishing feature giving the algebraic computation models additional power is the ability to make finitely many tests, usually either equality or order. Both examples are Weihrauch equivalent to LPO. Thus we are lead to the problem of classifying the computational power inherent in being allowed to make finitely many uses of LPO. Note that we are

not required to state any bounds in advance (which would just yield LPO^*), but simply have to cease making additional queries to LPO eventually.

We can formalize this using the generalized register machines: We allow all computable functions on $\{0,1\}^{\mathbb{N}}$ as functions, and LPO as test[4].

Definition 9. Let $\mathrm{LPO}^\diamond :\subseteq \mathbb{N} \times (\{0,1\}^{\mathbb{N}})^* \to (\{0,1\}^{\mathbb{N}})^*$ take as input a Gödel-number of some generalized register machine M on $\{0,1\}^{\mathbb{N}}$ with computable functions (specified as part of the Gödel-number) and LPO-tests, as well as some input $(p_0, \ldots, p_n) \in (\{0,1\}^{\mathbb{N}})^*$ for such a machine. The output of $\mathrm{LPO}^\diamond$ is whatever M would output on input $(p_0, \ldots, p_n)$.

Of course, the preceding definition makes sense with some arbitrary multi-valued function f in place of LPO, and would give rise to an operation $^\diamond$ on the Weihrauch degrees. This operation is related to the *generalized Weihrauch reductions* proposed by Hirschfeldt and Jockusch in [55, 56]. While a detailed investigation of $^\diamond$ seems highly desirable, it is beyond the scope of the present paper and thus relegated to future work.

Proposition 10. $\mathrm{C}_{\mathbb{N}} \equiv_{\mathrm{W}} \mathrm{LPO}^\diamond$.

Proof. $\max_{\mathbb{N}^{\mathbb{N}}} \leq_{\mathbf{W}} \mathbf{LPO}^\diamond$ We describe a generalized register machine program using computable functions and LPO that solves $\max_{\mathbb{N}^{\mathbb{N}}}$. Given $p \in \mathbb{N}^{\mathbb{N}}$ and $n \in \mathbb{N}$, we can compute some $p_n \in \{0,1\}^{\mathbb{N}}$ such that $p_n \neq 0^{\mathbb{N}} \Leftrightarrow \exists i \in \mathbb{N} p(i) > n$. Starting with $i = 0$, we simply test $\mathrm{LPO}(p_i)$. If yes, we output i and terminate. Else we continue with $i := i + 1$.

$\mathbf{LPO}^\diamond \leq_{\mathbf{W}} \mathbf{C}_{\mathbb{N}}$ Consider some generalized register machine with computable functions and LPO-tests. Each computation path of the machine corresponding to some valid input is finite, in particular, uses LPO only finitely many times. We encode the results of the LPO-tests along such a finite path by a sequence of numbers $a_1, \ldots, a_m$ in the following way: If the result of the i^{th} test is 1, we put $a_i = 0$. If it is 0, we put $a_i = c + 1$, where c is a "precision parameter", intended to represent a bound on the occurrence of the first 1 in the input to LPO. Using a standard tupling function we can encode the sequence $a_1, \ldots, a_m$ into a single natural number $\langle a_1, \ldots, a_m \rangle$. Furthermore, we can arrange that every natural number encodes such a tuple.

Given a natural number $\langle a_1, \ldots, a_m \rangle$ which encodes the results of the LPO-tests along a finite path, we can check if the choices are infeasible. The number is rejected in three cases:

1. If the path does not end in a leaf.
2. If there exists $1 \leq i \leq m$ such that $a_i = 0$ but the input to the i^{th} LPO-test is non-zero.

[4] Alternatively, we could have used the rather cumbersome Oracle-Type-Two machines suggested in [39].

3. If $a_i = c+1$ but the input to the i^{th} LPO-test starts with more than c zeroes.

We can effectively enumerate all numbers that are rejected. This amounts to being able to compute the set $\mathrm{NR} \in \mathcal{A}(\mathbb{N})$ of numbers that are never rejected. We apply $\mathrm{C}_{\mathbb{N}}$ to this set to pick a number which is never rejected, which allows us to simulate the computation of the register machine in an otherwise computable fashion.

□

On a side note, let us consider two more computational models: First, the concept of finitely revising computation presented in [57] by Ziegler: These are Type-2 machines equipped with the additional power to reset their output finitely many times. It is easy to see that being computable by a finitely revising machine is equivalent to being Weihrauch-reducible to $\lim_{\Delta}$. Second, non-deterministic Type-2 computation, also introduced by Ziegler [58] and fleshed out further by Brattka, de Brecht and P. in [34]: Here the machine may guess an element of an advice space, and either proceed to successfully compute a solution, or reject the guess at a finite stage (and there must be a chance of the former). As shown in [34], being computable by a non-deterministic machine with advice space $\mathbf{Z}$ is equivalent to being Weihrauch reducible to $\mathrm{C}_{\mathbf{Z}}$. We thus arrive at the following:

Theorem 11. The following computational models are equivalent in the sense that they yield the same class of computable functions:

1. Generalized register machines with computable functions and LPO as test.
2. Finitely revising machines.
3. Non-deterministic machines with advice space $\mathbb{N}$.

We could equivalently have used generalized register machines over $\mathbb{R}$, with partial computable functions over $\mathbb{R}$ and $=$ as test. As BSS-machines are a restricted case of these, it is clear that simulating a BSS-machine is no harder than solving $\mathrm{LPO}^{\diamond}$.

3.2. BSS-machines over the reals

So far we have only obtained an upper bound for the power of BSS-machines in the Weihrauch lattice. For a lower bound, we require some further representatives of the Weihrauch degree of $\mathrm{C}_{\mathbb{N}}$. Let $\mathbb{Q}^{\mathrm{e}}_{+}$ denote the set of non-negative rational numbers understood as a subspace of the represented space $\mathbb{R}$, i.e. represented by the appropriate post-restriction of ρ. In contrast, let $\mathbb{Q}^{\mathrm{d}}_{+}$ be the discrete space of non-negative rational numbers, represented by $\delta_{\mathbb{Q}}(0^k 1 0^n 1 0^m 1^\omega) = \frac{n}{m+1}$. Some of the following have already been shown in [36, 35].

Proposition 12. The following are Weihrauch equivalent:

1. $\mathrm{C}_{\mathbb{N}}$.

2. $\max_{\mathbb{N}^{\mathbb{N}}} :\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$.
3. $\mathrm{id}^{\mathrm{e,d}}_{\mathbb{Q}_+} : \mathbb{Q}^{\mathrm{e}}_+ \to \mathbb{Q}^{\mathrm{d}}_+$.
4. Numerator : $\mathbb{Q}^{\mathrm{e}}_+ \to \mathbb{N}$, where $\mathrm{Numerator}(q) = n$ iff $\exists m \in \mathbb{N}\ \gcd(n, m) = 1$ and $|q| = \frac{n}{m}$.
5. Denominator : $\mathbb{Q}^{\mathrm{e}}_+ \rightrightarrows \mathbb{N}$, where $\mathrm{Denominator}(q) = m$ iff $\exists n \in \mathbb{N}\ \gcd(n, m) = 1$ and $|q| = \frac{n}{m}$.([5])

Proof. $\mathbf{C}_{\mathbb{N}} \leq_{\mathbf{W}} \max_{\mathbb{N}^{\mathbb{N}}}$ Lemma 3.

$\max_{\mathbb{N}^{\mathbb{N}}} \leq_{\mathbf{W}}$ Denominator W.l.o.g., assume that the input p to $\max_{\mathbb{N}^{\mathbb{N}}}$ is monotone and that $p(0) = 0$. We proceed to compute a non-negative real number x which will happen to be rational (i.e. we compute x as an element of $\mathbb{Q}^{\mathrm{e}}_+$). Our initial approximation to x is $x_0 = 0$. If $p(n+1) = p(n)$, then $x_{n+1} = x_n$. If $p(n+1) > p(n)$, then we search for some $k, l \in \mathbb{N}$ such that $d\left(\frac{2l+1}{2^{\langle k, p(n+1)\rangle}}, x_n\right) < 2^{-n-1}$ – which are guaranteed to exist. Then we set $x_{n+1} = \frac{2l+1}{2^{\langle k, p(n+1)\rangle}}$. As the range of p is finite, this sequence will stabilize eventually, and by construction, converges quickly to its rational limit.

Applying Denominator to x will give us some $2^{\langle k, \max_{\mathbb{N}^{\mathbb{N}}}(p)\rangle} \in \mathbb{N}$. By design of $\langle\ ,\ \rangle$, we can extract $\max_{\mathbb{N}^{\mathbb{N}}}(p)$ from this value.

$\max_{\mathbb{N}^{\mathbb{N}}} \leq_{\mathbf{W}}$ Numerator Very similar to the reduction $\max_{\mathbb{N}^{\mathbb{N}}} \leq_{\mathrm{W}}$ Denominator. On monotone input p with $p(0) = 0$, we start with the approximation $x_0 = 1$. If $p(n+1) = p(n)$, then $x_{n+1} = x_n$. Otherwise, we search for $k, l \in \mathbb{N}$ such that $d\left(\frac{2^{\langle p(n+1), k\rangle}}{2l+1}, x_n\right) < 2^{-n-1}$, and set $x_{n+1} := \frac{2^{\langle p(n+1), k\rangle}}{2l+1}$ for these values. As the range of p is finite, this sequence will stabilize eventually, and by construction, converges quickly to its rational limit x.

Applying Numerator to x will give us some $2^{\langle k, \max_{\mathbb{N}^{\mathbb{N}}}(p)\rangle} \in \mathbb{N}$. By design of $\langle\ ,\ \rangle$, we can extract $\max_{\mathbb{N}^{\mathbb{N}}}(p)$ from this value.

Denominator $\leq_{\mathbf{W}} \mathbf{id}^{\mathbf{e,d}}_{\mathbb{Q}_+}$ Trivial.

Numerator $\leq_{\mathbf{W}} \mathbf{id}^{\mathbf{e,d}}_{\mathbb{Q}_+}$ Trivial.

$\mathbf{id}^{\mathbf{e,d}}_{\mathbb{Q}_+} \leq_{\mathbf{W}} \mathbf{C}_{\mathbb{N}}$ Given a non-negative real number $x \in \mathbb{R}_+$, we can compute the closed set $\{\langle n, m\rangle \in \mathbb{N} \mid mx = n, m \neq 0\} \in \mathcal{A}(\mathbb{N})$. If $x \in \mathbb{Q}^{\mathrm{e}}_+$ this set is non-empty, so we can use $\mathrm{C}_{\mathbb{N}}$ to extract an element, which allows us to obtain $x \in \mathbb{Q}^{\mathrm{d}}_+$.

□

Proposition 13. $\mathrm{id}^{\mathrm{e,d}}_{\mathbb{Q}_+}$ is computable by a machine over $(\mathbb{R}, +, =, 1)$.

[5] This map is multivalued, as any positive integer is a valid output on input 0. Restricting the map to positive inputs does not change the Weihrauch degree.

Proof. Let x be the input. We can test if $x = x + x$, in which case we know that $x = 0$. If this is not the case we compute for all pairs $n, m \in \mathbb{N}$ with $n, m \geq 1$, the numbers mx and n by repeated addition and test them for equality. If they are equal, then we have found a valid output in the pair (n, m), if not, we consider the next pair. □

Corollary 14. For every algebraic computation model over every structure expanding $(\mathbb{R}, +, =, 1)$ not exceeding the computable functions and $\{=, <\}$ as tests, we find that:

- Every partial function computable in that model is Weihrauch reducible to $\mathrm{C}_{\mathbb{N}}$.
- There is a (partial) function computable in that model that is Weihrauch equivalent to $\mathrm{C}_{\mathbb{N}}$.

Thus, the computational power of algebraic computation models is, from the perspective of topological computation models, characterized by the Weihrauch degree of $\mathrm{C}_{\mathbb{N}}$.

3.3. BSS-machines which compute total functions

It should be pointed out though that the characterisation in Corollary 14 relies crucially on considering partial functions, too. For total functions, we obtain instead a family of upper bounds as follows:

Definition 15 ([59])**.** Let $R \subseteq \mathbb{N} \times \mathbb{N}$ be a well-founded partial order. We define $\mathfrak{L}_R :\subseteq (\mathbb{N} \times \mathbb{N}^{\mathbb{N}})^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ as follows: A sequence $(n_i, x_i)_{i \in \mathbb{N}}$ where $n_i \in \mathbb{N}$ and $x_i \in \mathbb{N}^{\mathbb{N}}$ is in the domain of $\mathfrak{L}_R$, if $x_i \neq x_{i+1} \Rightarrow (n_{i+1}, n_i) \in R$ and $x_i = x_{i+1} \Rightarrow n_i = n_{i+1}$. As R is well-founded, these conditions imply that the sequence stabilizes eventually, and $\mathfrak{L}_R$ returns the corresponding limit x_∞.

Theorem 16 (Computable Hausdorff-Kuratowski theorem [60])**.** For a total function $f : \mathbb{R}^* \to \mathbb{R}^*$ the following are equivalent:

1. $f \leq_{\mathrm{W}} \mathrm{C}_{\mathbb{N}}$.
2. There exists some computable R such that $f \leq_{\mathrm{W}} \mathfrak{L}_R$.

Let $R \subseteq \mathbb{N} \times \mathbb{N}$ and $P \subseteq \mathbb{N} \times \mathbb{N}$ be two well-founded partial orders such that there exists an order-preserving map f from R to P, i.e. some $f : \mathbb{N} \to \mathbb{N}$ such that if $(n, m) \in R$, then $(f(n), f(m)) \in P$. It follows that $\mathfrak{L}_R \leq^c_{\mathrm{W}} \mathfrak{L}_P$. Conversely, results from [61, 59] show that this implication indeed reverses. Consider well-founded partial orders up to the equivalence notion induced by the existence of order-preserving maps in both directions is one construction of the countable ordinals – and as shown in [60], it can indeed be seen as a canonical one. Thus, for any countable ordinal α we can associate a continuous Weihrauch degree $\mathfrak{L}_\alpha$ as the degree of $\mathfrak{L}_R$ for any/every well-founded partial order R with rank α.

Proposition 17 ([59])**.** For countable ordinals $\alpha < \beta$ we have $\mathfrak{L}_\alpha <^c_{\mathrm{W}} \mathfrak{L}_\beta <^c_{\mathrm{W}} \mathrm{C}_{\mathbb{N}}$.

Corollary 18. Let $f : \mathbb{R}^* \to \mathbb{R}^*$ be a total BSS-computable function. Then there is some countable ordinal α with $f \leq^c_{\mathrm{W}} \mathfrak{L}_\alpha$.

Proposition 19. For each countable successor ordinal $\alpha+1$ there is some BSS-computable total function $f_{\alpha+1} : [0,1] \to \mathbb{N}$ (using constants) with $f_{\alpha+1} \geq^c_{\mathrm{W}} \mathfrak{L}_{\alpha+1}$.

Proof. We proceed by induction over $\alpha+1$. The claim for $\alpha = 0$ is witnessed by LPO. It suffices to show that if the claim is true (uniformly) for $(\alpha_i)_{i\in\mathbb{N}}$ with $\alpha_i \leq \alpha_{i+1}$, then it is true for $(\sup_{i\in\mathbb{N}} \alpha_i) + 1 =: \alpha + 1$.

We define $f_{\alpha+1} : [0,1] \to \mathbb{N}$ piecewise. If $x \in [\frac{1}{2i+1}, \frac{1}{2i}]$ for some $i \in \mathbb{N}$, then $f_{\alpha+1}(x) := \langle i+1, f_{\alpha_i+1}(2i(2i+1)x - 2i)\rangle$. Otherwise, $f_{\alpha+1}(x) := \langle 0,0\rangle$. If the f_{α_i+1} are either uniformly (in i) BSS-computable, or, alternatively, coded into a real parameter, this clearly yields a BSS-computable function.

Claim: $\mathfrak{L}_{\alpha+1} \leq^c_{\mathbf{W}} f_{\alpha+1}$

We start writing a name for $x := 0$ while reading the input to $\mathfrak{L}_{\alpha+1}$. If the first parameter ever changes, then it will move to some n_i such that the lower cone of n_i has some height $\alpha' + 1 \leq \alpha$. In particular, there must be some $j \in \mathbb{N}$ with $\alpha' + 1 \leq \alpha_j + 1$. The suitable pairs (i,j) can be coded into a single real parameter. There will be some $k \geq j$ such that $[\frac{1}{2k+1}, \frac{1}{2k}]$ is still within the scope of the current approximation to 0.

The tail of the input to $\mathfrak{L}_{\alpha+1}$ at the current moment is now also a valid input to $\mathfrak{L}_{\alpha_k+1}$. Thus, we can continue to use the reduction $\mathfrak{L}_{\alpha_k+1} \leq^c_{\mathrm{W}} f_{\alpha_k+1}$, with f_{α_k+1} scaled down into the interval $[\frac{1}{2k+1}, \frac{1}{2k}]$.

To interpret the output of $f_{\alpha+1}$, the only difference is whether it is of the form $\langle 0,0\rangle$ or $\langle i+1, n\rangle$. In the former case, the input sequence to $\mathfrak{L}_{\alpha+1}$ is constant, and we just read off the answer. In the latter case, there is at least one change – so we can search for it, and then split $n := \langle i', n'\rangle$ to see whether there is a further change, and so on.

□

3.4. BSS-machines without order tests

If we consider total functions, and drop the order test from the signature, the maximum Weihrauch degree reachable is even lower. This is caused by two properties of algebraic sets that are not shared with semi-algebraic sets: every descending chain of algebraic sets eventually stabilises and every proper algebraic subset of an irreducible variety V is nowhere dense. These properties will cause any computation tree of an algorithm which computes a total function to be finite.

Proposition 20. If $f : \mathbb{R}^* \to \mathbb{R}^*$ is BSS-computable over $(\mathbb{R}, +, \times, =)$, then $f \leq^c_{\mathrm{W}} \mathrm{LPO}^*$. There is a function $g : \mathbb{R}^* \to \mathbb{R}^*$ that is BSS-computable over $(\mathbb{R}, +, \times, =)$ and satisfies $g \equiv_{\mathrm{W}} \mathrm{LPO}^*$.

Proof. Let $f\colon \mathbb{R}^* \to \mathbb{R}^*$ be BSS-computable over $(\mathbb{R}, +, \times, =)$. Fix some BSS-machine computing f. For a fixed input dimension n, consider the computation tree of that machine. We claim that the tree is finite. It then follows that

$f \leq_{\mathrm{W}}^{c} \mathrm{LPO}^*$, for we can bound the number of equality tests we need to simulate the machine in terms of the size of the input tuple alone.

If the computation tree is infinite it has an infinite path by Kőnig's lemma. Each node on the path corresponds to an algebraic set, and the outgoing edge from each node is labelled "$\in$" or "$\notin$", depending on whether we branch on the condition that the input is in the algebraic set or outside of it. Since the Zariski topology on $\mathbb{R}^n$ is Noetherian, there are only finitely many edges labelled with "$\in$" in a non-trivial way. By this we mean the following: if we number the nodes on the path with numbers $0, 1, 2, \ldots$ then there exists a number $N \in \mathbb{N}$ such that for any node v on the path which is labelled with a number greater than N and whose outgoing edge on the path is labelled with "$\in$", the computation on any input will branch to "$\in$" if it reaches this node. Let V be the algebraic set which is defined by the nodes with number smaller than N whose outgoing edge on the path is labelled with "$\in$". Let $(V_n)_n$ be the sequence of algebraic sets which correspond to the nodes whose outgoing edge is labelled with "$\notin$". Our goal is to show that there exists $x \in \mathbb{R}^n$ with $x \in V$ and $x \notin V_n$ for all $n \in \mathbb{N}$. This means that x passes all tests on the infinite branch (since all tests which do not correspond to V or one of the V_n's are passed automatically), which means that the machine runs forever on input x, contradicting the totality of f. The set V is a finite union of irreducible algebraic sets, at least one of which is not contained in any of the V_n's. We can hence assume without loss of generality that V is itself an irreducible algebraic variety. The sets $V_n \cap V$ are (potentially empty) proper algebraic subsets of V and thus have dense open complement in the Euclidean topology on V. By the Baire category theorem, their countable union has dense open complement in V. In particular, this complement contains a point. This finishes the proof.

For the example g, consider the function $g : \mathbb{R}^* \to \mathbb{N}$ mapping an input tuple $(x_1, \ldots, x_n)$ to the set $|\{i \in \{1, \ldots, n\} \mid x_i = 0\}|$. This function is obviously BSS-computable over $(\mathbb{R}, +, \times, =)$, and easily seen to be Weihrauch-equivalent to LPO^*. □

4. On the BSS-Halting problems

Very much in analogy to the (classical) Halting problem and the investigation of the Turing degrees below it (Post's problem), the BSS-Halting problems are easily seen to be undecidable by the corresponding BSS-machines, and there are rich hierarchies to be found below them [62, 63]. Very much unlike the classical setting, there is a natural undecidable set strictly below the Halting problem, namely $\mathbb{Q}$.

Let $\mathbb{H} \subseteq \mathbb{N} \times \mathbb{R}^*$ be the Halting problem for BSS-machines having access to $+$, $=$, 1 and potentially additional computable operations and/or $<$ as test[6].

[6]Of course, changing the signature changes the set $\mathbb{H}$, but as the proof of Theorem 21 is independent of these details, they do not change the Weihrauch degree of $\chi_{\mathbb{H}}$.

This means that $(n, x_0, \ldots, x_m) \in \mathbb{H}$ iff n is a Gödel-number for a BSS-machine that on input $(x_0, \ldots, x_m)$ will eventually halt.

Let $\chi_{\mathbb{Q}} : \mathbb{R} \to \{0,1\}$ be the characteristic function of $\mathbb{Q}$ (into the discrete space $\{0,1\}$), and $\chi_{\mathbb{H}} : \mathbb{N} \times \mathbb{R}^* \to \{0,1\}$ be the characteristic function of $\mathbb{H}$. Finally, let $\mathsf{isInfinite} : \{0,1\}^{\mathbb{N}} \to \{0,1\}$ be defined via $\mathsf{isInfinite}(p) = 1$ iff $|\{n \in \mathbb{N} \mid p(n) = 1\}| = \infty$.

Theorem 21. The following are Weihrauch-equivalent:

1. $\chi_{\mathbb{Q}} : \mathbb{R} \to \{0,1\}$
2. $\chi_{\mathbb{H}} : \mathbb{N} \times \mathbb{R}^* \to \{0,1\}$
3. $\mathsf{isInfinite} : \{0,1\}^{\mathbb{N}} \to \{0,1\}$

Proof. $\chi_{\mathbb{Q}} \leq_{\mathbf{W}} \chi_{\mathbb{H}}$ Note that $\chi_{\mathbb{Q}} \equiv_{\mathrm{W}} \chi_{\mathbb{Q}_+}$, where $\mathbb{Q}_+$ denotes the set of non-negative rational numbers. Now, $\chi_{\mathbb{Q}_+} \leq_{\mathrm{W}} \chi_{\mathbb{H}}$ is easily established: we just combine the original input with the program for $\mathrm{id}^{\mathrm{e,d}}_{\mathbb{Q}_+}$ from Proposition 13. This will halt iff the input is rational.

$\chi_{\mathbb{H}} \leq_{\mathbf{W}} \mathsf{isInfinite}$ Given the Gödel number of a BSS-machine M and a standard name p of a point $x \in \mathbb{R}^*$, we construct a sequence $(A_n)_n$ of Type-2 algorithms as follows: The n^{th} algorithm simulates the machine M on input p until it reaches an equality- or order-test. It then tries to show that the result of the test is "false" with precision parameter n (cf. the proof of Proposition 10). Otherwise it assumes that the result of the test is "true" and continues to simulate M in this manner. If $n < m$, we say that A_m *refutes* A_n *within* l *steps* if A_n and A_m take different branches on the equality- or order-tests within l steps of computation. This defines a relation between the numbers m, n, and l which in general depends on p (and not just on x). Note that this relation is decidable relative to p.

Now consider the following Type-2 algorithm: Assign a variable $n = 0$. For all pairs $(m, l) \in \mathbb{N}^2$ do the following: Run the machine A_n for l steps. If it doesn't halt within those l steps, write a 1 on the output tape. Test if A_m refutes A_n within l steps. If so, put $n = m$ and write a 1 on the output tape. Finally, write a 0 on the output tape, and continue looping.

We claim that the sequence this algorithm produces contains finitely many 1s if and only if M halts on input x. Assume that the sequence contains finitely many 1s. Then there exists $n \in \mathbb{N}$ such that A_n halts within a finite number of steps and no A_m refutes A_n in any finite number of steps. But this means that A_n correctly simulates M on input x, for if it erroneously decides a test to be "true", it will eventually be refuted. Hence, M halts on input x within a finite number of steps. Conversely, if M halts on input x, then it makes only finitely many equality- or order-tests before halting. Hence for sufficiently large n the algorithm A_n correctly simulates M on input x, and thus is never refuted and halts after finitely many steps. It follows that our algorithm only writes finitely many 1s on the output tape.

Hence we can decide if M halts on input x by applying $\mathsf{isInfinite}$ to the output of the algorithm.

isInfinite $\leq_{\mathbf{W}} \chi_{\mathbb{Q}}$ Compute the real number with the decimal expansion

$$0.a_1 0 a_2 0 0 a_3 0 0 0 a_4 0 0 0 0 \ldots$$

where a_i is the ith bit of the input. This number is rational, i.e. has a periodic decimal expansion, if and only if the a_i are eventually 0.

□

This shows that the role of (local) cardinality for BSS-reducibility demonstrated in [64] depends on the set of operations available in the reduction. Further, note that $\mathsf{isInfinite} \not\leq^{c}_{\mathrm{W}} \mathrm{C}_{\mathbb{N}}$ is easily seen: While $\mathrm{C}_{\mathbb{N}}$ is Δ^0_2-measurable (cf. e.g. [42]), isInfinite is the characteristic function of a Π^0_2-complete set, thus not even Σ^0_2-measurable. By [43], levels of Borel measurability are preserved under Weihrauch reductions. Thus, the inability of BSS-machines to decide their Halting problem already holds for topological reasons (in particular, adding more continuous operations never allows a machine to solve a more restricted Halting problem).

As a side note, we shall point out that isInfinite also is the degree of deciding whether a Type-2 computable function is well-defined on some particular input:

Proposition 22. Let $\mathsf{isDefined} : \mathbb{N}\times\{0,1\}^{\mathbb{N}} \to \{0,1\}$ be defined via $\mathsf{isDefined}(n,p) = 1$ iff the n-th Type-2 machine produces some $q \in \{0,1\}^{\mathbb{N}}$ on input p. Then:

$$\mathsf{isDefined} \equiv_{\mathrm{W}} \mathsf{isInfinite}$$

Proof. Let s be an index for the Type-2 machine that copies each 1 from the input to the output, and skips all 0s. Then $\mathsf{isInfinite}(p) = \mathsf{isDefined}(s,p)$.

For the other direction, note that we can simulate the n-th Type-2 machine while writing 0s. Whenever the machine outputs something, we write a 1. Applying isInfinite to the output produces an answer for isDefined. □

5. The Weihrauch degree of sorting and other problems

In this section we will investigate some Weihrauch degrees, in particular the degree of sorting some $p \in \{0,1\}^{\mathbb{N}}$ by order. This degree will turn out to be crucial in characterizing the power of strongly analytic machines later.

Let $\mathfrak{a} = (a_n)_{n\in\mathbb{N}}$ be a computable, infinite, repetition-free and dense sequence in the complete computable metric space $\mathbf{X}$. Let $\mathrm{Type}_{\mathfrak{a}} : \mathbf{X} \to [0,1]$ be defined via $\mathrm{Type}_{\mathfrak{a}}(a_n) = 2^{-n}$ and $\mathrm{Type}_{\mathfrak{a}}(x) = 0$ if $x \notin \mathrm{range}(\mathfrak{a})$.

Let $\mathsf{Sort} : \{0,1\}^{\mathbb{N}} \to \{0,1\}^{\mathbb{N}}$ be defined via $\mathsf{Sort}(p) = 0^n 1^\omega$ iff p contains exactly n times the bit 0, and $\mathsf{Sort}(p) = 0^\omega$ iff p contains infinitely many 0s.

Theorem 23. $\mathsf{Sort} \equiv_{\mathrm{W}} \mathrm{Type}_{\mathfrak{a}}$ for any $\mathfrak{a}$.

Proof. $\mathrm{Type}_{\mathfrak{a}} \leq_{\mathrm{W}} \mathsf{Sort}$

Let x be the input to $\mathrm{Type}_{\mathfrak{a}}$. Start testing if $x = a_0$. While this is possible, write 1's on the input to Sort. If $x \neq a_0$ is ever proven, write a single 0 and proceed to test if $x = a_1$ instead, while again writing 1's. Repeat indefinitely.

The outer reduction witness is given by computable $K :\subseteq \{0,1\}^{\mathbb{N}} \times \{0,1\}^{\mathbb{N}} \to [0,1]$ with $K(p, 0^n 1^\omega) = 2^{-n}$ and $K(p, 0^\omega) = 0$.

$\mathsf{Sort} \leq_{\mathrm{W}} \mathrm{Type}_{\mathfrak{a}}$

As long as we find only 1's in the input to Sort, start writing a_0 as the input x to $\mathrm{Type}_{\mathfrak{a}}$. If a 0 is read (at time t), we have specified x by fixing $x \in B(a_0, 2^{-t})$. As the $(a_n)_{n\in\mathbb{N}}$ are dense, we can compute an injective function $f : \mathbb{N} \to \mathbb{N}$ such that $(a_{f(n)})_{n\in\mathbb{N}}$ ranges over all a_n in $B(a_0, 2^{-t})$ (and we may assume that $f(0) = 0$). We proceed to write approximations to $a_{f(1)}$ while we read 1's on the input to Sort. If the next 0 is read, we again compute a suitable subsequence and switch our approximations to the next element, and so on.

Given the output of $\mathrm{Type}_{\mathfrak{a}}$, we can start by deciding whether it is 1 or not. If it is 1, the output of Sort must be 1^ω. If not, then the input to Sort must contain some 0, and we can search until we find the first one at position t. Knowing t means we can recover the computable function f constructed in the inner reduction witness. Then we test whether the output of $\mathrm{Type}_{\mathfrak{a}}$ is $2^{-f(1)}$. If so, the output of Sort is 01^ω. If not, there is a second 0 somewhere in the input at time t', etc. □

Let $\mathsf{isFinite}_{\mathbb{S}} : \{0,1\}^{\mathbb{N}} \to \mathbb{S}$ be defined via $\mathsf{isFinite}_{\mathbb{S}}(p) = \top$ iff p contains finitely many 1s. Let $\mathsf{isInfinite}_{\mathbb{S}} : \{0,1\}^{\mathbb{N}} \to \mathbb{S}$ be defined via $\mathsf{isInfinite}_{\mathbb{S}}(p) = \top$ iff p contains infinitely many 1s. Let $\mathrm{TC}_{\mathbb{N}} : \mathcal{A}(\mathbb{N}) \rightrightarrows \mathbb{N}$ be the total continuation of $\mathrm{C}_{\mathbb{N}}$, i.e. $p \in \mathrm{TC}_{\mathbb{N}}(A)$ iff $p \in A \vee A = \emptyset$.

Proposition 24.

1. $\mathrm{C}_{\mathbb{N}} <_{\mathrm{W}} \mathsf{Sort} <_{\mathrm{W}} \widehat{\mathsf{Sort}} \equiv_{\mathrm{W}} \lim$
2. $\mathsf{isInfinite}_{\mathbb{S}} <_{\mathrm{W}} \mathrm{TC}_{\mathbb{N}}$
3. $\mathsf{isFinite}_{\mathbb{S}} <_{\mathrm{W}} \mathsf{Sort}$
4. $\mathsf{isInfinite}_{\mathbb{S}} \not\leq_{\mathrm{W}} \mathsf{Sort}$
5. $\mathsf{isFinite}_{\mathbb{S}} \not\leq_{\mathrm{W}} \mathrm{TC}_{\mathbb{N}}$

Proof.

[Claim: $\mathbf{C_{\mathbb{N}} \leq_W Sort}$] We show $\max_{\mathcal{O}} \leq_{\mathrm{W}} \mathsf{Sort}$ instead and appeal to Lemma 3. Given some set $A \in \mathcal{O}(\mathbb{N})$, we start writing 1's. In addition, we make sure that we write exactly as many 0s as the largest number encountered in A so far. If A is guaranteed to be finite, then the resulting sequence contains only finitely many 0's, and after it was sorted, we can read off how many there are – which returns $\max_{\mathcal{O}} A$.

[Claim: $\mathbf{Sort \leq_W \lim}$] Given some $p \in \{0,1\}^{\mathbb{N}}$, let $q(\langle n, i\rangle) = 0$ if $p_{\leq i}$ contains at least n 0s, and $q(\langle n, i\rangle) = 1$ otherwise. Then $\mathsf{Sort}(p) = \lim(q)$.

[Claim: $\mathbf{\widehat{Sort} \leq_W \lim}$] From $\mathsf{Sort} \leq_{\mathrm{W}} \lim$ with $\widehat{\lim} \equiv_{\mathrm{W}} \lim$.

[Claim: $\mathbf{\lim \leq_W \widehat{Sort}}$] From $\mathrm{C}_{\mathbb{N}} \leq_{\mathrm{W}} \mathsf{Sort}$ and $\lim \equiv_{\mathrm{W}} \widehat{\mathrm{C}_{\mathbb{N}}}$ (see [34, Example 3.10]).

[**Claim:** $\lim \not\leq_{\mathbf{W}}$ **Sort**] Sort maps every input to some computable output. lim maps some computable inputs to non-computable outputs.

[**Claim:** $\mathbf{isInfinite}_{\mathbb{S}} <_{\mathbf{W}} \mathbf{TC}_{\mathbb{N}}$] Given $p \in \{0,1\}^{\mathbb{N}}$, we can compute the set

$$\{n \in \mathbb{N} \mid p \text{ contains no more than } n \text{ 1 s}\} \in \mathcal{A}(\mathbb{N}).$$

Apply $\mathrm{TC}_{\mathbb{N}}$ to obtain some $m \in \mathbb{N}$. Now read p while writing 0s. If ever the $(m+1)$-st 1 in p is found, then the input to $\mathrm{TC}_{\mathbb{N}}$ must have been the empty set, i.e. p must contain infinitely many 1s. Switching the output to writing 1s from then on causes the output to be correct.

That the reduction is strict follows from the fact that $\mathrm{C}_{\mathbb{N}} \leq_{\mathrm{W}} \mathrm{TC}_{\mathbb{N}}$ (by definition), whereas $\mathsf{isInfinite}_{\mathbb{S}}$ has a codomain with just 2 elements.

[**Claim:** $\mathbf{isFinite}_{\mathbb{S}} <_{\mathbf{W}}$ **Sort**] Let $S : \{0,1\}^{\mathbb{N}} \to \{0,1\}^{\mathbb{N}}$ be the map that swaps 0s and 1s. Then $\mathsf{isFinite}_{\mathbb{S}}(p) = \delta_{\mathbb{S}} \circ \mathsf{Sort} \circ S$.

As above, that the reduction is strict follows from the fact that $\mathrm{C}_{\mathbb{N}} \leq_{\mathrm{W}} \mathsf{Sort}$ as shown above, whereas $\mathsf{isFinite}_{\mathbb{S}}$ has a codomain with just 2 elements.

[**Claim:** $\mathbf{isInfinite}_{\mathbb{S}} \not\leq_{\mathbf{W}}$ **Sort**] As $\mathsf{Sort} \leq_{\mathrm{W}} \lim$, we find that Sort is Σ^0_2-measurable, and so is every function reducible to it (cf. [43]). But $\mathrm{isInfinite}_{\mathbb{S}}^{-1}(\{\top\})$ is a Π^0_2-complete set.

[**Claim:** $\mathbf{isFinite}_{\mathbb{S}} \not\leq_{\mathbf{W}} \mathbf{TC}_{\mathbb{N}}$] Assume that $\mathsf{isFinite}_{\mathbb{S}} \leq_{\mathrm{W}} \mathrm{TC}_{\mathbb{N}}$ via some witnesses K', H. By composing K' with $\delta_{\mathbb{S}}$ and $\delta_{\mathbb{N}}^{-1}$, we obtain computable $K :\subseteq \{0,1\}^{\mathbb{N}} \times \mathbb{N} \to \mathbb{S}$. We can assume K to be total, as we can let it write 0 infinitely many times without changing the value of the output. Note that we can assume that this procedure for making K total preserves single-valuedness since $\mathbb{N}$ is discrete. We can turn H into a computable function $h : \{0,1\}^* \to \mathbb{N}^*$ such that $n \notin \psi(H(p))$ iff $\exists l.\ (n+1) \in h(p_{\leq l})$, where $p_{\leq l}$ is the prefix of p of length l. We will reason with K and h in the following.

As $\mathsf{isFinite}_{\mathbb{S}}(0^\omega) = \top$, there have to be $l_0, k \in \mathbb{N}$ such that $K(0^{l_0}\{0,1\}^\omega, k) = \top$. As $\mathsf{isFinite}_{\mathbb{S}}(0^{l_0}1^\omega) = \bot$, there has to be some $l_1 \in \mathbb{N}$ such that $h(0^{l_0}1^{l_1}) \ni k+1$.

Now we proceed in stages $i \in \mathbb{N}$, each associated with some current prefix $p_i \in \{0,1\}^*$. We start with $i = 0$ and $p_0 = 0^{l_0}1^{l_1}$. In stage i, consider $K(p_i 0^\omega, i)$. If this is $\bot$, then we must have $i \notin \psi(H(p_i 0^\omega))$ (otherwise the reduction could answer $\bot$ wrongly), so there is some j such that $h(p_i 0^j) \ni i+1$. We set $p_{i+1} := p_i 0^j 1$ and continue with the next stage. If $K(p_i 0^\omega, i) = \top$, then this is already determined by some finite prefix $p_i 0^j$. Hence, we must have that $i \notin \psi(H(p_i 0^j 1^\omega))$ (otherwise the reduction could answer $\top$ wrongly), so there is some m with $h(p_i 0^j 1^m) \ni i+1$. We set $p_{i+1} := p_i 0^j 1^m 1$ and continue with the next stage.

We find that $p = \lim_{i\to\infty} p_i \in \{0,1\}^{\mathbb{N}}$ is well-defined, contains infinitely many 1s and satisfies that $\psi(H(p)) = \emptyset$. Thus, a realiser of $\mathrm{TC}_{\mathbb{N}}$ may

answer anything on input $H(p)$, in particular it may answer k. But $K(p,k) = \top$, so the reduction fails.

[**Claim: $\mathsf{Sort} \not\leq_{\mathbf{W}} \mathbf{C}_{\mathbb{N}}$**] By combining $\mathsf{isFinite}_{\mathbb{S}} <_{\mathrm{W}} \mathsf{Sort}$ and $\mathsf{isFinite}_{\mathbb{S}} \not\leq_{\mathrm{W}} \mathrm{TC}_{\mathbb{N}}$, and noting that $\mathrm{C}_{\mathbb{N}} \leq_{\mathrm{W}} \mathrm{TC}_{\mathbb{N}}$ by definition.

□

We next wish to show that $\mathsf{Sort} <_{\mathrm{W}} \mathsf{Sort}^2 <_{\mathrm{W}} \mathsf{Sort}^3 <_{\mathrm{W}} \ldots$. For this, we need a slight generalization of the Squashing Theorem from [65] (cf. the development in [66]). Rather than using the notion of a finitely-tolerant function as employed there, we generalize this to weakly finitely-tolerant functions. The proof remains unaffected by this, though.

Definition 25. Call $f : \mathbb{N}^{\mathbb{N}} \rightrightarrows \mathbb{N}^{\mathbb{N}}$ weakly finitely-tolerant, if there is a computable function A such that for any $\lambda, \lambda' \in \mathbb{N}^*$, $p, q \in \mathbb{N}^{\mathbb{N}}$, if $q \in f(\lambda p)$, then $A(q, \lambda, \lambda') \in f(\lambda' p)$.

Theorem 26 (Squashing Theorem [65])**.** If $f : \mathbb{N}^{\mathbb{N}} \rightrightarrows \mathbb{N}^{\mathbb{N}}$ is weakly finitely tolerant and $f \equiv_{\mathrm{W}} f \times f$, then $f \equiv_{\mathrm{W}} \widehat{f}$.

Corollary 27. For any $n \in \mathbb{N}$, $\mathsf{Sort}^n <_{\mathrm{W}} \mathsf{Sort}^{n+1}$.

Proof. It is easy to see that any Sort^i is weakly finitely tolerant. Thus, if the claim were false, the Squashing Theorem would imply $\mathsf{Sort}^i \equiv_{\mathrm{W}} \widehat{\mathsf{Sort}} \equiv_{\mathrm{W}} \lim$ (from Proposition 24 (1)), but the left-hand side has only computable outputs, whereas the right-hand side maps some computable inputs to non-computable outputs. □

Proposition 28. 1. $\mathsf{Sort} \star \mathrm{LPO} \equiv_{\mathrm{W}} \mathsf{Sort} \times \mathrm{LPO}$
2. $\mathsf{Sort} \star \mathrm{C}_{\mathbb{N}} \equiv_{\mathrm{W}} \mathsf{Sort} \times \mathrm{C}_{\mathbb{N}}$

Proof. Note that we can extend any computable partial function $f :\subseteq \mathbb{N}^{\mathbb{N}} \to \{0,1\}^{\mathbb{N}}$ to a computable total function $F : \mathbb{N}^{\mathbb{N}} \to \{0,1\}^{\mathbb{N}}$ such that $\mathsf{Sort} \circ f(p) = \mathsf{Sort} \circ F(p)$ for any $p \in \mathrm{dom}(f)$ – just add infinitely many 1's to the (partial) output of f. Furthermore, for both cases it suffices to show the $\leq_{\mathrm{W}}$-direction, the $\geq_{\mathrm{W}}$-direction trivially holds.

1. As long as the input to LPO is consistent with $0^{\mathbb{N}}$, we use the corresponding input to Sort. If we ever read a 1 in the input to LPO, we restart writing the input to Sort corresponding to the now known output of LPO. By looking at the output of LPO (on the right hand side), we can determine in which case we are. In the former, the output of Sort already is correct. In the latter, we can find out the precise finite prefix of the input to Sort we had written when encountering the 1 in the input to LPO, count the 0s in that prefix and adjust the output of Sort accordingly.

2. We use $\max_{\mathbb{N}^\mathbb{N}}$ instead of $\mathrm{C}_\mathbb{N}$. The argument proceeds similar as above: Start by providing the input to Sort that would correspond to $\max_{\mathbb{N}^\mathbb{N}}$ outputting 0. Once we learn that $\max_{\mathbb{N}^\mathbb{N}}$ will provide a larger value, switch to the corresponding input to Sort. Repeat as required. We can then use $\max_{\mathbb{N}^\mathbb{N}}$ (on the right hand side) to determine the length of the finite wrong prefix fed to Sort, and change the output of Sort accordingly to fix it.

□

Proposition 29. $\mathrm{TC}_\mathbb{N} \leq_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathsf{isFinite}_\mathbb{S}$ and $\mathsf{Sort} \leq_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathsf{isInfinite}_\mathbb{S}$.

Proof. **First claim**: Given some $A \subseteq \mathbb{N}$, we can compute $p_A \in \{0,1\}^\mathbb{N}$ such that $|\{k \in \mathbb{N} \mid p_A(k) = 1\}| \geq n$ iff $\{0, \ldots, n-1\} \cap A = \emptyset$. Apply $\mathsf{isFinite}_\mathbb{S}$ to this, and then $(\mathrm{id} : \mathbb{S} \to \{0,1\}) \equiv_\mathrm{W} \mathrm{LPO}$ to the output. If the answer is 1, the original input is a valid input for $\mathrm{C}_\mathbb{N}$. If we learn 0 from the first part, feed a name for $\mathbb{N}$ to $\mathrm{C}_\mathbb{N}$. The output of $\mathrm{C}_\mathbb{N}$ yields a solution to $\mathrm{TC}_\mathbb{N}$ in either case. Thus, $\mathrm{TC}_\mathbb{N} \leq_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathrm{LPO} \star \mathsf{isFinite}_\mathbb{S} \equiv_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathsf{isFinite}_\mathbb{S}$. Here we used $\mathrm{LPO} \leq_\mathrm{W} \mathrm{C}_\mathbb{N}$, and $\mathrm{C}_\mathbb{N} \star \mathrm{C}_\mathbb{N} \equiv_\mathrm{W} \mathrm{C}_\mathbb{N}$ (as recalled in Lemma 6).

Second claim: We show that $\mathsf{Sort} \leq_\mathrm{W} \max_{\mathbb{N}^\mathbb{N}} \star \mathrm{LPO} \star \mathsf{isInfinite}_\mathbb{S}$ instead. Let $S : \{0,1\}^\mathbb{N} \to \{0,1\}^\mathbb{N}$ swap 0 and 1 componentwise, and let $p \in \{0,1\}^\mathbb{N}$ be the original input to Sort. Apply $\mathrm{id} : \mathbb{S} \to \{0,1\}$ to the output of $\mathsf{isInfinite}_\mathbb{S}$ on input$S(p)$. If we receive a 1, feed $0^\mathbb{N}$ to max, and answer $0^\mathbb{N}$ for Sort. Else, define $q \in \mathbb{N}^\mathbb{N}$ by $q(n) = |\{k \leq n \mid p(k) = 0\}|$, and note that $q \in \mathrm{dom}(\max_{\mathbb{N}^\mathbb{N}})$. Then $0^{\max q} 1^\omega$ is the correct output to Sort. □

Corollary 30. $\mathrm{C}_\mathbb{N} \star \mathsf{Sort} \equiv_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathsf{isFinite}_\mathbb{S} \equiv_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathsf{isInfinite}_\mathbb{S} \equiv_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathrm{TC}_\mathbb{N}$

Proof. $\mathbf{C}_\mathbb{N} \star \mathbf{isInfinite}_\mathbb{S} \leq_\mathbf{W} \mathbf{C}_\mathbb{N} \star \mathbf{TC}_\mathbb{N}$ By Proposition 24 (2).

$\mathbf{C}_\mathbb{N} \star \mathbf{TC}_\mathbb{N} \leq_\mathbf{W} \mathbf{C}_\mathbb{N} \star \mathbf{isFinite}_\mathbb{S}$ By Proposition 29 we have $\mathrm{C}_\mathbb{N} \star \mathrm{TC}_\mathbb{N} \leq_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathrm{C}_\mathbb{N} \star \mathsf{isFinite}_\mathbb{S} \equiv_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathsf{isFinite}_\mathbb{S}$, invoking Lemma 6.

$\mathbf{C}_\mathbb{N} \star \mathbf{isFinite}_\mathbb{S} \leq_\mathbf{W} \mathbf{C}_\mathbb{N} \star \mathbf{Sort}$ By Proposition 24 (3).

$\mathbf{C}_\mathbb{N} \star \mathbf{Sort} \leq_\mathbf{W} \mathbf{C}_\mathbb{N} \star \mathbf{isInfinite}_\mathbb{S}$ By Proposition 29 we have $\mathrm{C}_\mathbb{N} \star \mathsf{Sort} \leq_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathrm{C}_\mathbb{N} \star \mathsf{isInfinite}_\mathbb{S} \equiv_\mathrm{W} \mathrm{C}_\mathbb{N} \star \mathsf{isInfinite}_\mathbb{S}$.

□

Our next goal is to show that Sort is 2-low, in the sense that $\lim \star \lim \star \mathsf{Sort} \equiv_\mathrm{W} \lim \star \lim$. We start with a more general result, for which we need the notion of a precomplete represented space. A space $\mathbf{Y}$ is called precomplete, if for every partial computable function $F :\subseteq \{0,1\}^\mathbb{N} \to \mathbf{Y}$ there is a total computable function $F' : \{0,1\}^\mathbb{N} \to \mathbf{Y}$ such that $F = F'|_{\mathrm{dom}(F)}$. Typical examples of precomplete spaces are $\mathbb{S}$, $\mathcal{O}(\mathbb{N})$ and $\mathcal{A}(\mathbb{N})$.

We further use the precomplete space $\mathbb{S}_{\Sigma^0_2}$ with underlying set $\{\bot, \top\}$ and representation $\delta_{\Sigma^0_2} : \{0,1\}^\mathbb{N} \to \{\bot, \top\}$ defined via $\delta_{\Sigma^0_2}(p) = \top$ iff p contains infinitely many 1s, and $\delta_{\Sigma^0_2}(p) = \bot$ else. Now the map $\mathrm{id} : \mathbb{S}_{\Sigma^0_2} \to \{0,1\}$ (mapping $\bot$ to 0 and $\top$ to 1) has the same realizer as $\mathsf{isInfinite}$. Moreover, $\widehat{\mathsf{isInfinite}} \equiv_\mathrm{W} \lim \star \lim$ was shown in [43].

Proposition 31. Let $g : \mathbf{X} \rightrightarrows \mathbb{N}$, and $f : \mathbf{Y} \rightrightarrows \mathbf{Z}$ where $\mathbf{Y}$ is precomplete. Then $f \star g \leq_{\mathrm{W}} \widehat{f} \times g$.

Proof. We make use of the explicit representative of $f \star g$ constructed in [29]. The input of $f \star g$ is a partial continuous function $e :\subseteq \mathbb{N} \rightrightarrows \mathbb{N}^{\mathbb{N}} \times \mathbf{Y}$ and some $x \in \mathbf{X}$. The output is a pair $(p, z) \in \mathbb{N}^{\mathbb{N}} \times \mathbf{Z}$ such that there exists $y \in \mathbf{Y}$ such that $(p, y) \in e(g(x))$ and $z \in f(y)$. As $\mathbb{N}$ has an injective representation, we can assume w.l.o.g. that e is actually a single-valued (partial) function, which then splits into $e_1 :\subseteq \mathbb{N} \to \mathbb{N}^{\mathbb{N}}$ and $e_2 :\subseteq \mathbb{N} \to \mathbf{Y}$. As $\mathbf{Y}$ is precomplete, we can extend e_2 to a total function, and by currying, obtain a sequence $(y_0, y_1, \ldots)$. We apply $\widehat{f}$ to $(y_0, y_1, \ldots)$ and obtain some sequence $(z_0, z_1, \ldots,)$, and g to x to obtain $n \in \mathbb{N}$. The pair $(e_1(n), z_n)$ constitutes a valid output for $f \star g$. □

Corollary 32. $\lim \star \lim \star \mathsf{Sort} \equiv_{\mathrm{W}} \lim \star \lim$

Proof. From Proposition 29 we can in particular conclude that $\mathsf{Sort} \leq_{\mathrm{W}} \mathrm{C}_{\mathbb{N}} \star \mathsf{isInfinite}$, and the righthand side has up to isomorphism codomain $\mathbb{N}$. The Weihrauch degree of $\lim \star \lim$ has $\widehat{\left(\mathrm{id} : \mathbb{S}_{\Sigma^0_2} \to \{0,1\}\right)}$ as a representative with precomplete domain. Thus, from Proposition 31 we conclude:

$$\lim \star \lim \star \mathsf{Sort} \leq_{\mathrm{W}} (\lim \star \lim) \times (\mathrm{C}_{\mathbb{N}} \star \mathsf{isInfinite})$$

Since $(\mathrm{C}_{\mathbb{N}} \star \mathsf{isInfinite}) \leq_{\mathrm{W}} \mathrm{C}_{\mathbb{N}} \star (\mathrm{LPO} \star \mathsf{isFinite}_{\mathbb{S}}) \leq_{\mathrm{W}} (\mathrm{C}_{\mathbb{N}} \star \mathrm{LPO}) \star \mathsf{Sort} \leq_{\mathrm{W}} \lim \star \lim$ using Proposition 24 (1,3), Lemma 6 (1), and Lemma 5 (2), the claim follows. □

Corollary 33. $\coprod_{n \in \mathbb{N}} \mathsf{Sort}^{(n)} \leq_{\mathrm{W}} \lim \star \lim$.

Proof. By iterating Corollary 32 n times, we find that:

$$\lim \star \lim \star \mathsf{Sort}^{(n)} \leq_{\mathrm{W}} \lim \star \lim$$

As this argument is uniform in n, the claim follows. □

6. The algebraic decision problem

We are now ready to introduce and study a canonical problem associated with strongly analytic machines. Let $(P_{n,d})_{n \in \mathbb{N}}$ be some standard enumeration of the d-variate polynomials with rational coefficients.

Definition 34. Define functions $\textsc{AlgDec} : \mathbb{R}^* \to [0, 1]$ and $\textsc{AlgDec}_d : \mathbb{R}^d \to [0, 1]$ via

$$\textsc{AlgDec}_d(x_1, \ldots, x_d) = \sum_{\{n | P_{n,d}(x_1, \ldots, x_d) = 0\}} 2^{-2n-2}$$

and $\textsc{AlgDec}(x_1, \ldots, x_m) = \textsc{AlgDec}_m(x_1, \ldots, x_m)$.

The choice for $[0,1]$ as the codomain for ALGDEC is just to ease the comparison to functions computable by strongly analytic machines, we could just as well have defined $\text{ALGDEC}_d : \mathbb{R}^d \to \{0,1\}^{\mathbb{N}}$ with $\text{ALGDEC}_d((x_1,\ldots,x_d))(n) = 1$ iff $P_{n,d}(x_1,\ldots,x_d) = 0$. Thus, intuitively ALGDEC will tell us the "algebraic type" (in the sense of the definition before Theorem 23) of the input tuple.

Observation 35. If $f : \mathbb{R}^* \to \mathbb{R}^*$ is computable by a strongly analytic machine, then $f \leq_{\text{W}} \text{ALGDEC}$. Moreover, ALGDEC is computable by a strongly analytic machine.

Proof. That ALGDEC is computable by a strongly analytic machine is immediate. For the remainder of the claim, we argue that if a strongly analytic machine M computes $f : \mathbb{R}^* \to \mathbb{R}^*$ on input $\mathbf{x}$, then a Type-2 machine can simulate M if provided $\mathbf{x}$ and $\text{ALGDEC}(\mathbf{x})$ as input. Clearly, the only obstacle to such a simulation are the equality tests that M can make. Each of these is of the form $p(\mathbf{x}) = 0?$, where p is a rational multivariate polynomial[7]. If $p = P_{n,d}$, then inspecting $\text{ALGDEC}(\mathbf{x})$ up to precision 2^{-2n-4} allows the Type-2 machine to determine whether or not $p(\mathbf{x}) = 0$. □

Observation 36. $\text{ALGDEC} \equiv_{\text{W}} (\coprod_{d\in\mathbb{N}} \text{ALGDEC}_d)$

Proposition 37. $\text{ALGDEC}_1 \equiv_{\text{W}} \mathsf{Sort}$

Proof. Let $\mathfrak{a}$ be an effective enumeration of the algebraic numbers in $\mathbb{R}$. We understand this to mean that an index n of an algebraic number a_n encodes the minimal polynomial of a_n, together with some information about which root (e.g. ordered by $<$) a_n is of its minimal polynomial. By Theorem 23 we have that $\mathsf{Sort} \equiv_{\text{W}} \text{TYPE}_{\mathfrak{a}}$, thus we only need to show $\text{ALGDEC}_1 \equiv_{\text{W}} \text{TYPE}_{\mathfrak{a}}$.

For $\text{ALGDEC}_1 \leq_{\text{W}} \text{TYPE}_{\mathfrak{a}}$ we show that for a given rational polynomial P the predicate $P(x) = 0$ is decidable relative to $\text{TYPE}_{\mathfrak{a}}(x)$. Given a non-zero rational polynomial P, we verify in parallel if $P(x) \neq 0$ and if $\text{TYPE}_{\mathfrak{a}}(x) \neq 0$. Clearly, one of the searches has to terminate. If the second search terminates, it yields the minimal polynomial of x. Now we can decide if $P(x) = 0$ by deciding if the minimal polynomial divides P.

For $\text{TYPE}_{\mathfrak{a}} \leq_{\text{W}} \text{ALGDEC}_1$, observe that from any non-zero rational polynomial P and a real number $x \in \mathbb{R}$ with $P(x) = 0$, we can compute the minimal polynomial of x and determine the position of x in the list of its roots. □

Theorem 38. $\text{ALGDEC}_d \equiv_{\text{W}} \text{ALGDEC}_1^d$

Corollary 39. $\text{ALGDEC} \equiv_{\text{W}} \mathsf{Sort}^*$

In order to prove Theorem 38 we need to recall a few facts from (computational) commutative algebra. Let $\mathcal{I}(\mathbb{Q}[X_1,\ldots,X_d])$ denote the represented space of ideals in $\mathbb{Q}[X_1,\ldots,X_d]$, where an ideal is represented by some finite set of generators (that this is a representation follows from Hilbert's basis theorem).

[7] If M is using real constants, we would consider these as part of $\mathbf{x}$.

Recall that the *height* $\mathrm{ht}(P)$ of a prime ideal P is the length n of the longest chain of strict inclusions

$$P = P_n \supsetneq P_{n-1} \supsetneq \cdots \supsetneq P_1 \supsetneq P_0,$$

where the P_i's are prime ideals. The *Krull dimension* of a ring R is the supremum of the heights of all prime ideals in R. If K is a field, then the polynomial ring $K[X_1, \ldots, X_d]$ has Krull dimension d. We will need the following well-known facts from computer algebra (see e.g. [67, 68, 69, 70, 71])

Fact 40.

1. Membership of a polynomial $f \in \mathbb{Q}[X_1, \ldots, X_d]$ in an ideal $I \in \mathcal{I}(\mathbb{Q}[X_1, \ldots, X_d])$ is decidable.
2. Primality of a given ideal $I \in \mathcal{I}(\mathbb{Q}[X_1, \ldots, X_d])$ is decidable.
3. The height of a given prime ideal $P \in \mathcal{I}(\mathbb{Q}[X_1, \ldots, X_d])$ is computable. □

Proof of Theorem 38. The direction $\textsc{AlgDec}_1^d \leq_{\mathrm{W}} \textsc{AlgDec}_d$ is trivial. For the converse direction we prove $\textsc{AlgDec}_d \leq_W \mathsf{Sort}^d$ and apply Proposition 37. For a given point $x \in \mathbb{R}^d$, consider the prime ideal $I(x) = \{f \in \mathbb{Q}[X_1, \ldots, X_d] \mid f(x) = 0\}$. Our goal is to compute the characteristic function of $I(x)$. We will use Sort^d to approximate $I(x)$ "from below" in the following sense: we will compute a sequence $(P_n)_n$ of prime ideals with $P_n \subseteq I(x)$ for all n and $P_n = I(x)$ for sufficiently large n. Using the sequence $(P_n)_n$ we can verify if $f \in I(x)$ by searching for an $n \in \mathbb{N}$ such that $f \in P_n$, using Fact 40 (1). Conversely, we can verify if $f \notin I(x)$ by verifying if $f(x) \neq 0$.

It remains to construct the sequence $(P_n)_n$. By Fact 40 we can compute for each $h \in [1; d]$ a sequence $(P_{h,n})_n$ containing all prime ideals in $\mathbb{Q}[X_1, \ldots, X_d]$ of height $\geq h$. For each of these sequences we use an instance of Sort to compute a new sequence $(P'_{h,n})_n$ with $P'_{h,n} \subseteq I(x)$, proceeding like in the proof of the reduction $\mathrm{Type}_{\mathfrak{a}} \leq_W \mathsf{Sort}$ in Theorem 23: start with $P_{h,0}$ and try to prove that $P_{h,0} \not\subseteq I(x)$ by searching for a generator f of $P_{h,0}$ with $f(x) \neq 0$. At the same time, write 1's to the input of Sort. If $P_{h,0} \not\subseteq I(x)$ is proved, write a 0 and continue with $P_{h,1}$. Apply Sort to the resulting sequence to obtain a new sequence $p \in \{0,1\}^{\mathbb{N}}$. If $p = 0^N 1^\omega$, put

$$P'_{h,n} = \begin{cases} \langle 0 \rangle & \text{if } n \leq N, \\ P_{h,N} & \text{otherwise.} \end{cases}$$

If $p = 0^\omega$, put $P'_{h,n} = \langle 0 \rangle$ for all $n \in \mathbb{N}$.

By construction, each of the ideals $P'_{h,n}$ is contained in $I(x)$. If $h \leq \mathrm{ht}(I(x))$, then there exists $n \in \mathbb{N}$ such that $P'_{h,n}$ has height $\geq h$. In particular, if $h = \mathrm{ht}(I(x))$, then there exists $P'_{h,n} \subseteq I(x)$ with $\mathrm{ht}(P'_{h,n}) \geq \mathrm{ht}(I(x))$. Since $P'_{h,n}$ is prime it follows that $P'_{h,n} = I(x)$. Since $\mathbb{Q}[X_1, \ldots, X_d]$ has Krull dimension d, the height $\mathrm{ht}(I(x))$ is a number between 0 and d, and if $\mathrm{ht}(I(x)) = 0$, then $I(x) = \langle 0 \rangle$, so that $P'_{h,n} = I(x)$ for all $n \in \mathbb{N}$, $h \in [1; d]$. In any case, there always exist h and n such that $P'_{h,n} = I(x)$. Using standard coding tricks we

may write the double-sequence $(P'_{h,n})_{h\in[1;d],n\in\mathbb{N}}$ as a single sequence $(J_n)_n$. The J_n's are a sequence of prime ideals contained in $I(x)$, at least one of which is equal to $I(x)$. Now put $P_n = \sum_{k=0}^{n} J_k$. Then $P_n \subseteq I(x)$ for all n and $P_n = I(x)$ for sufficiently large n. □

[15, Question 3.9] asks whether there is a set $A \subseteq \mathbb{R}$ which is *weakly semidecidable*, yet not a Π^0_3-set. They define a set to be weakly semidecidable, if it is BSS many-one reducible to the boundedness problem for analytic machines. This in turn means that there is a BSS-computable function $H : \mathbb{R} \to \mathbb{R}^*$, and an analytic machine that on input $H(x)$ for $x \in A$ computes some bounded sequence $(a_i) \in \mathbb{R}^{\mathbb{N}}$, and on input $H(x)$ for $x \notin A$ computes some unbounded sequence $(a_i) \in \mathbb{R}^{\mathbb{N}}$. We can give a negative answer:

Proposition 41. Every weakly semidecidable set is Δ^0_3.

Proof. Given some weakly semidecidable set A, we want to provide an upper bound on the Weihrauch degree of $\chi_A : \mathbb{R} \to \{0,1\}$. By combining Corollary 14, Observation 35 and Corollary 39, we see that there is a function $a : \mathbb{R} \to \mathbb{R}^{\mathbb{N}}$ with $a \leq_{\mathrm{W}} \mathsf{Sort}^* \star \mathrm{C}_{\mathbb{N}}$, such that $a(x)$ is bounded iff $x \in A$.

Now given $(a_i) \in \mathbb{R}^{\mathbb{N}}$, we can compute some $p \in \{0,1\}^{\mathbb{N}}$ such that p contains at least n 0s iff $\exists k\ |a_k| > n$. Thus, using $\mathsf{isInfinite}$, we can detect whether a real sequence is bounded or unbounded.

Put together, we conclude that $\chi_A \leq_{\mathrm{W}} \mathsf{isInfinite}\star\mathsf{Sort}^*\star\mathrm{C}_{\mathbb{N}}$. Since $\mathsf{isInfinite} \leq_{\mathrm{W}} \lim\star\lim$, we have $\chi_A \leq_{\mathrm{W}} \lim\star\lim\star\mathsf{Sort}^* \star \mathrm{C}_{\mathbb{N}}$. By Corollary 32 the righthand side is reducible to $\lim\star\lim\star\mathrm{C}_{\mathbb{N}}$. With Lemma 6 we conclude $\chi_A \leq_{\mathrm{W}} \lim\star\lim$. As $\lim\star\lim$ is Σ^0_3-measurable, the claim follows. □

7. Comparing the SCI in the two models

Following [16, 17] we shall define the *solvability complexity index* over the BSS-model and over the TTE-model, and then use the results on Weihrauch degrees obtained in the preceding sections to bound their difference.

Definition 42. An n-tower for a function $f : \mathbb{R}^* \to \mathbb{R}^*$ is a function $F : \mathbb{N}^n \times \mathbb{R}^* \to \mathbb{R}^*$ such that $f(x_1,\ldots,x_m) = \lim_{i_1\to\infty}\ldots\lim_{i_n\to\infty} F(i_1,\ldots,i_n,x_1,\ldots,x_m)$. For some function $f : \mathbb{R}^* \to \mathbb{R}^*$, let $\mathrm{SCI}_{\mathrm{BSS}}(f)$ be the least n such that there a BSS-computable n-tower for f. Let $\mathrm{SCI}_{\mathrm{TTE}}(f)$ be the least n such that there a computable (i.e. TTE-computable) n-tower for f.

Observation 43. $\mathrm{SCI}_{\mathrm{TTE}}(f) \leq n$ iff $f \leq_{\mathrm{W}} \lim^{(n)}$.

Theorem 44. If $\mathrm{SCI}_{\mathrm{TTE}}(f) \geq 2$ or $\mathrm{SCI}_{\mathrm{BSS}}(f) \geq 2$, then $\mathrm{SCI}_{\mathrm{TTE}}(f) = \mathrm{SCI}_{\mathrm{BSS}}(f)$.

Proof. **Assume $\mathbf{SCI_{TTE}}(f) = n \geq 1$.** Let $F : \mathbb{N}^n \times \mathbb{R}^* \to \mathbb{R}^*$ be a computable n-tower for f. By the Stone-Weierstrass Approximation Theorem, we can approximate F by rational polynomials on each hypercube, i.e. there are rational multivariate polynomials $g^k_{i_1,\ldots,i_n}$ such that for $(x_1,\ldots,x_m) \in [-k,k]^m$ we find that $d(g^k_{i_1,\ldots,i_n}(x_1,\ldots,x_m), F(i_1,\ldots,i_n,x_1,\ldots,x_m)) < 2^{-k}$. As we

can code a computable countable sequence of rational multivariate polynomials into a computable parameter, we find that $G : \mathbb{N}^n \times \mathbb{R}^* \to \mathbb{R}^*$ defined via $G(i_1, \ldots, i_n, x_1, \ldots, x_m) = g^{i_n}_{i_1,\ldots,i_n}(x_1, \ldots, x_m)$ is BSS-computable. Moreover, it is straight-forward to verify that G also is an n-tower for f. Thus, $\mathrm{SCI}_{\mathrm{TTE}}(f) \geq \mathrm{SCI}_{\mathrm{BSS}}(f)$.

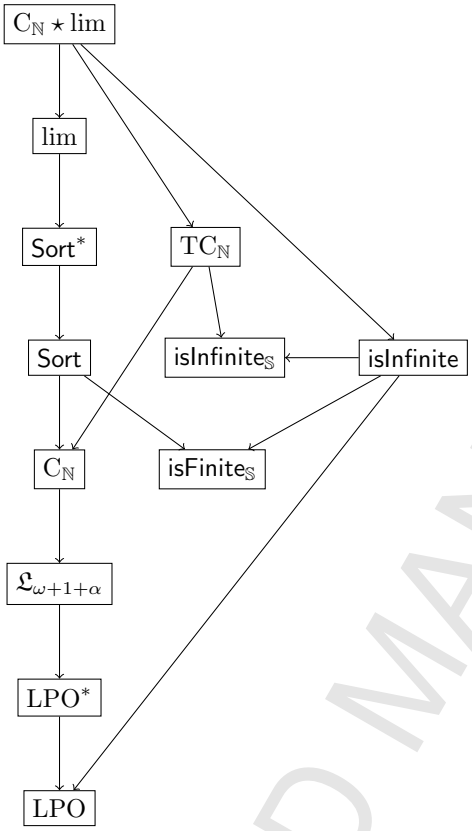
Assume $\mathbf{SCI_{BSS}}(f) = n \geq 2$. Let F be a BSS-computable n-tower for f. We curry F to obtain $G : \mathbb{R}^* \to (\mathbb{R}^*)^{\mathbb{N}^n}$, and notice that with the same reasoning as for strongly analytic functions in Section 6, we find that $G \leq_{\mathrm{W}} \mathsf{Sort}^*$. By assumption, we then have that $f \leq_{\mathrm{W}} \lim^{(n)} \star\mathsf{Sort}^*$. As $n \geq 2$, Corollary 32 implies that already $f \leq_{\mathrm{W}} \lim^{(n)}$. By [32, Fact 5.5] lim is a *transparent cylinder*, i.e. it follows that there is a computable function H such that $f = \lim \circ \ldots \circ \lim \circ H$. But this means that H is a computable n-tower for f, i.e. $\mathrm{SCI}_{\mathrm{BSS}}(f) \geq \mathrm{SCI}_{\mathrm{TTE}}(f)$. □

Corollary 45. For $n \geq 2$, $\mathrm{SCI}_{\mathrm{BSS}}(f) \geq n$ iff $f \not\leq_{\mathrm{W}} \lim^{(n-1)}$.

Theorem 44 provides a formal version of the informal idea that a function that is *very non-computable* in the BSS-model is still so in the TTE-model and vice versa. This provides a reason to continue the investigation of the non-computability of an interesting function beyond establishing it – as a more precise classification can potentially be translated to the other setting via Theorem 44. This also shows that there is common ground between the two frameworks, and that this common ground includes the SCI (provided it is at least 2).

8. Summary diagram

The following diagram provides an overview of some of the relevant Weihrauch degrees. Arrows denote reductions in the reverse direction. The diagram is complete up to transitivity, i.e. if no arrow is present in the transitive closure of the diagram, then there is a separation proof for the principles.



- lim captures limit computability.
- Sort* captures computability by strongly analytic machines
- isInfinite contains BSS-Halting problems
- $C_{\mathbb{N}}$ captures computability by BSS-machines.
- LPO* captures computability of total functions by BSS-machines without $<$

References

[1] A. Grzegorcyk, Computable functionals, Fundamenta Mathematicae 42 (1955) 168–202.

[2] A. Grzegorcyk, On the definition of computable real continuous functions, Fundamenta Mathematicae 44 (1957) 61–71.

[3] D. Lacombe, Extension de la notion de fonction récursive aux fonctions d'une ou plusieurs variables réelles III, Comptes Rendus Académie des Sciences Paris 241 (1955) 151–153.

[4] K. Weihrauch, Computability, Monographs on Theoretical Computer Science, Springer-Verlag, 1987.

[5] K. Weihrauch, Computable Analysis, Springer-Verlag, 2000.

[6] M. Pour-El, I. Richards, Computability in analysis and physics, Perspectives in Mathematical Logic, Springer, 1989.

[7] K.-I. Ko, Computational Complexity of Real Functions, Birkhäuser, 1991.

[8] L. Blum, M. Shub, S. Smale, On a theory of computation and complexity over the real numbers: NP- completeness, recursive functions and universal machines, Bull. Amer. Math. Soc. 21 (1) (1989) 1 – 46.

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

[10] M. Escardó, Synthetic topology of datatypes and classical spaces, Electronic Notes in Theoretical Computer Science 87.

[11] A. Pauly, On the topological aspects of the theory of represented spaces, Computability 5 (2) (2016) 159–180. doi:10.3233/COM-150049.
URL http://arxiv.org/abs/1204.3763

[12] T. Chadzelek, G. Hotz, Analytic machines, Theoretical Computer Science 219 (1999) 151–167.

[13] G. Hotz, G. Vierke, B. Schieffer, Analytic machines, Tech. Rep. 25, Electronic Colloquium on Computational Complexity (1995).

[14] T. Gärtner, G. Hotz, Computability of analytic functions with analytic machines, in: K. Ambos-Spies, B. Löwe, W. Merkle (Eds.), Mathematical Theory and Computational Practice, Vol. 5635 of Lecture Notes in Computer Science, Springer, 2009, pp. 250–259. doi:10.1007/978-3-642-03073-4_26.
URL http://dx.doi.org/10.1007/978-3-642-03073-4_26

[15] T. Gärtner, M. Ziegler, Real analytic machines and degrees, Logical Methods in Computer Science 7.

[16] A. C. Hansen, On the solvability complexity index, the n-pseudospectrum and approximations of spectra of operators, Journal of the AMS 24 (2011) 81–124.

[17] J. Ben-Artzi, A. C. Hansen, O. Nevanlinna, M. Seidel, Can everything be computed? – on the solvability index and towers of algorithms, Comptes Rendus Mathematique 353 (10) (2015) 931–936.
URL http://arxiv.org/abs/1508.03280

[18] C. Gaßner, On NP-completeness for linear machines, Journal of Complexity 13 (2) (1997) 259 – 271. doi:10.1006/jcom.1997.0444.
URL http://www.sciencedirect.com/science/article/pii/S0885064X97904441

[19] C. Gaßner, The P-DNP problem for infinite abelian groups, Journal of Complexity 17 (3) (2001) 574 – 583. doi:10.1006/jcom.2001.0583.
URL http://www.sciencedirect.com/science/article/pii/S0885064X01905837

[20] C. Gaßner, On relativizations of the P =? NP question for several structures, Electronic Notes in Theoretical Computer Science 221 (2008) 71 – 83. doi:10.1016/j.entcs.2008.12.008.
URL http://www.sciencedirect.com/science/article/pii/S1571066108004714

[21] N. Tavana, K. Weihrauch, Turing machines on represented sets, a model of computation for analysis, Logical Methods in Computer Science 7 (2011) 1–21. doi:10.2168/LMCS-7(2:19)2011.

[22] J. Tucker, J. Zucker, Computable functions and semicomputable sets on many-sorted algebras, in: T. M. S. Abramsky, D.M. Gabbay (Ed.), Handbook of Logic in Computer Science, Vol. 5 of Oxford Science Publications, 2000, pp. 317–523.

[23] A. Hemmerling, Computability of string functions over algebraic structures, Mathematical Logic Quarterly 44 (1) (1998) 1–44. doi:10.1002/malq.19980440102.
URL http://dx.doi.org/10.1002/malq.19980440102

[24] M. Ziegler, Real computability and hypercomputation, Habilitationsschrift, University of Paderborn (2007).

[25] V. Brattka, G. Gherardi, A. Pauly, Weihrauch complexity in computable analysis, arXiv 1707.03202 (2017).

[26] A. Pauly, On the (semi)lattices induced by continuous reducibilities, Mathematical Logic Quarterly 56 (5) (2010) 488–502. doi:10.1002/malq.200910104.

[27] V. Brattka, G. Gherardi, Weihrauch degrees, omniscience principles and weak computability, Journal of Symbolic Logic 76 (2011) 143 – 176, arXiv:0905.4679.

[28] K. Higuchi, A. Pauly, The degree-structure of Weihrauch-reducibility, Logical Methods in Computer Science 9 (2). doi:10.2168/LMCS-9(2:2)2013.

[29] V. Brattka, A. Pauly, On the algebraic structure of Weihrauch degrees, arXiv 1604.08348 (2016).
URL http://arxiv.org/abs/1604.08348

[30] A. Pauly, How incomputable is finding Nash equilibria?, Journal of Universal Computer Science 16 (18) (2010) 2686–2710. doi:10.3217/jucs-016-18-2686.

[31] V. Brattka, G. Gherardi, Effective choice and boundedness principles in computable analysis, Bulletin of Symbolic Logic 1 (2011) 73 – 117, arXiv:0905.4685. doi:10.2178/bsl/1294186663.

[32] V. Brattka, G. Gherardi, A. Marcone, The Bolzano-Weierstrass Theorem is the jump of Weak König's Lemma, Annals of Pure and Applied Logic 163 (6) (2012) 623–625, also arXiv:1101.0792. doi:10.1016/j.apal.2011.10.006.

[33] V. Brattka, S. Le Roux, A. Pauly, On the computational content of the Brouwer fixed point theorem, in: S. Cooper, A. Dawar, B. Löwe (Eds.), How the World Computes, Vol. 7318 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 56–67. doi:10.1007/978-3-642-30870-3_7.

[34] V. Brattka, M. de Brecht, A. Pauly, Closed choice and a uniform low basis theorem, Annals of Pure and Applied Logic 163 (8) (2012) 968–1008. doi:10.1016/j.apal.2011.12.020.

[35] A. Pauly, Methoden zum Vergleich der Unstetigkeit von Funktionen, Masters thesis, FernUniversität Hagen (2007).

[36] U. Mylatz, Vergleich unstetiger Funktionen in der Analysis, Diplomarbeit, Fachbereich Informatik, FernUniversität Hagen (1992).

[37] U. Mylatz, Vergleich unstetiger Funktionen : "principle of omniscience" und Vollständigkeit in der C-Hierarchie, Ph.D. thesis, Fernuniversität, Gesamthochschule in Hagen (Mai 2006).

[38] V. Brattka, G. Gherardi, R. Hölzl, Probabilistic computability and choice, Information and Computation 242 (2015) 249 – 286, arXiv 1312.7305. doi:10.1016/j.ic.2015.03.005.
URL http://www.sciencedirect.com/science/article/pii/S0890540115000206

[39] A. Pauly, Infinite oracle queries in type-2 machines (extended abstract), arXiv:0907.3230v1 (July 2009).

[40] A. Pauly, G. Davie, W. Fouché, Weihrauch-completeness for layerwise computability, arXiv:1505.02091 (2015).

[41] A. Pauly, F. Steinberg, Comparing representations for function spaces in computable analysis, Theory of Computing Systems (2017) 1–26doi:10.1007/s00224-016-9745-6.
URL http://dx.doi.org/10.1007/s00224-016-9745-6

[42] A. Pauly, M. de Brecht, Non-deterministic computation and the Jayne Rogers theorem, Electronic Proceedings in Theoretical Computer Science 143, dCM 2012.

[43] V. Brattka, Effective Borel measurability and reducibility of functions, Mathematical Logic Quarterly 51 (1) (2005) 19–44. doi:10.1002/malq.200310125.

[44] H. de Holanda Cunha Nobrega, Game characterizations of function classes and Weihrauch degrees, M.Sc. thesis, University of Amsterdam (2013).

[45] K. Weihrauch, The TTE-interpretation of three hierarchies of omniscience principles, Informatik Berichte 130, FernUniversität Hagen, Hagen (Sep. 1992).

[46] D. S. Bridges, F. Richman, Varieties of Constructive Mathematics, Vol. 57 of Lecture Notes, London Mathematical Society, 1987.

[47] R. Penrose, The emperor's new mind, Oxford University Press, 1989.

[48] V. Brattka, The emperors new recursiveness: The epigraph of the exponential function in two models of computability, in: M. Ito, T. Imaoka (Eds.), Words, Languages & Combinatorics III, 2003, pp. 63–72.

[49] P. Hertling, Is the Mandelbrot set computable?, Mathematical Logic Quarterly 51 (1) (2005) 5–18. doi:10.1002/malq.200310124.
URL http://dx.doi.org/10.1002/malq.200310124

[50] M. Braverman, M. Yampolsky, Computability of Julia Sets, Springer, 2009.

[51] V. Brattka, P. Hertling, Feasible real random acess machines, Journal of Complexity 14 (1998) 490–526.

[52] N. Zhong, Recursively enumerable subsets of $\mathbb{R}^q$ in two computing models, Theoretical Computer Science 197 (1998) 79–94.

[53] P. Boldi, S. Vigna, Equality is a Jump, Theoretical Computer Science 219 (1999) 49–64.

[54] A. Morozov, M. Korovina, On Σ-definability without equality over the real numbers, Mathematical Logic Quarterly 54 (5) (2008) 535–544.

[55] D. Hirschfeldt, Slicing the Truth: On the Computability Theoretic and Reverse Mathematical Analysis of Combinatorial Principles, World Scientific, 2014.

[56] D. Hirschfeldt, C. Jockusch, On notions of computability-theoretic reduction between Π^1_2-principles, Journal of Mathematical Logic 16 (1).

[57] M. Ziegler, Revising type-2 computation and degrees of discontinuity, Electronic Notes in Theoretical Computer Science 167 (2007) 255–274. doi:10.1016/j.entcs.2006.08.015.

[58] M. Ziegler, Real hypercomputation and continuity, Theory of Computing Systems 41 (2007) 177 – 206. doi:10.1007/s00224-006-1343-6.

[59] M. de Brecht, Levels of discontinuity, limit-computability, and jump operators, in: V. Brattka, H. Diener, D. Spreen (Eds.), Logic, Computation, Hierarchies, de Gruyter, 2014, pp. 79–108, arXiv 1312.0697.

[60] A. Pauly, Computability on the countable ordinals and the Hausdorff-Kuratowski theorem, arXiv 1501.00386 (2015).

[61] P. Hertling, Unstetigkeitsgrade von Funktionen in der effektiven Analysis, Ph.D. thesis, Fernuniversität, Gesamthochschule in Hagen (Oktober 1996).

[62] K. Meer, M. Ziegler, An explicit solution to Post's problem over the reals, Journal of Complexity 24 (1) (2008) 3–15. doi:10.1016/j.jco.2006.09.004.
URL http://dx.doi.org/10.1016/j.jco.2006.09.004

[63] C. Gaßner, A hierarchy below the halting problem for additive machines, Theory of Computing Systems 43 (2008) 464–470.

[64] W. Calvert, K. Kramer, R. Miller, Noncomputable functions in the Blum-Shub-Smale model, Logical Methods in Computer Science 7 (2). doi:10.2168/LMCS-7(2:15)2011.

[65] F. G. Dorais, D. D. Dzhafarov, J. L. Hirst, J. R. Mileti, P. Shafer, On uniform relationships between combinatorial problems, Transactions of the AMS 368 (2016) 1321–1359, arXiv 1212.0157. doi:10.1090/tran/6465.

[66] T. Rakotoniaina, On the computational strength of Ramsey's theorem, Ph.D. thesis, University of Cape Town (2015).

[67] G.-M. Greuel, G. Pfister, A Singular Introduction to Commutative Algebra, 2nd Edition, Springer-Verlag, 2007.

[68] M. Kalkbrener, Algorithmic properties of polynomial rings, Journal of Symbolic Computation 26 (5) (1998) 525–582.

[69] P. Gianni, B. Trager, G. Zacharias, Gröbner bases and primary decomposition of polynomial ideals, Journal of Symbolic Computation 6 (2-3).

[70] B. Buchberger, Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal, Ph.D. thesis, University of Innsbruck, Austria (1965).

[71] B. Buchberger, Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems, Aequationes mathematicae 4 (3) (1970) 374–383.

Acknowledgements

This work was inspired by discussions at the workshop *Real Computation and BSS Complexity* in Greifswald, and the second author would like to thank the participants Russell Miller, Tobias Gärtner and Martin Ziegler, as well as the organizer Christine Gaßner. Moreover, the second author would like to thank Anders Hansen for fruitful discussions on the solvability complexity index.

The work presented here benefited from the Royal Society International Exchange Grant IE111233. The second author is partially supported by the ERC inVest project.