Monte Carlo Computability

Vasco Brattka[*1], **Rupert Hölzl**[2], **and Rutger Kuyper**[3]

1 **Department of Mathematics and Applied Mathematics, University of Cape Town, Cape Town, South Africa; and**
Faculty of Computer Science, Universität der Bundeswehr München, Neubiberg, Germany
Vasco.Brattka@cca-net.de

2 **Faculty of Computer Science, Universität der Bundeswehr München, Neubiberg, Germany**
r@hoelzl.fr

3 **School of Mathematics and Statistics, Victoria University of Wellington, Wellington, New Zealand**
mail@rutgerkuyper.com

Abstract

We introduce Monte Carlo computability as a probabilistic concept of computability on infinite objects and prove that Monte Carlo computable functions are closed under composition. We then mutually separate the following classes of functions from each other: the class of multi-valued functions that are non-deterministically computable, that of Las Vegas computable functions, and that of Monte Carlo computable functions. We give natural examples of computational problems witnessing these separations. As a specific problem which is Monte Carlo computable but neither Las Vegas computable nor non-deterministically computable, we study the problem of sorting infinite sequences that was recently introduced by Neumann and Pauly. Their results allow us to draw conclusions about the relation between algebraic models and Monte Carlo computability.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases Weihrauch degrees, Weak Weak Kőnig's Lemma, Monte Carlo computability, algorithmic randomness, sorting

Digital Object Identifier 10.4230/LIPIcs.STACS.2017.17

1 Introduction

It is folklore in computational complexity theory that different machine models such as deterministic machines, non-deterministic machines and probabilistic machines describe the same classes of computable problems while they potentially yield different classes of polynomial-time computable problems. The complexity classes NP, BPP and ZPP are classes of decision problems which are polynomial-time computable on non-deterministic machines, Monte Carlo machines, and Las Vegas machines, respectively. The separation of these classes is a major and challenging problem in computational complexity theory.

In recent years it emerged that the situation for computations on infinite objects is somewhat different in that the classes of problems which are non-deterministically computable or probabilistically computable are actually strictly larger than the class of deterministically

* Vasco Brattka has received funding from the National Research Foundation of South Africa. Rutger Kuyper has received funding from the John Templeton Foundation.

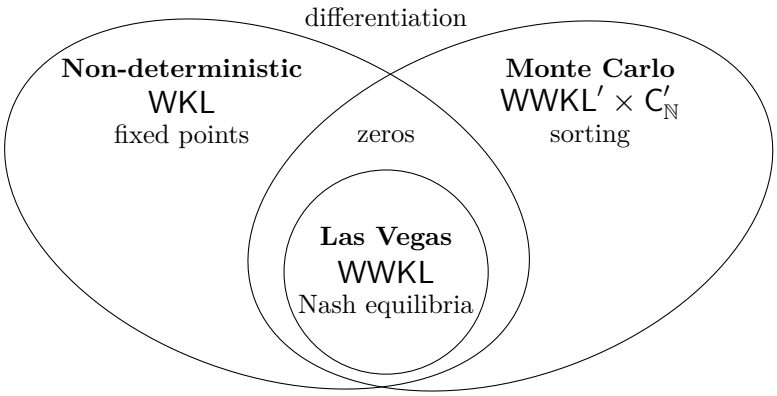

Figure 1 Classes of non-deterministically and probabilistically computable problems.

computable problems from the mere point of view of computability theory even without time complexity considerations.

Computability over infinite objects is understood here in the well-established sense of computable analysis [28, 13] and the Weihrauch lattice [6, 24] is used as a fine-grained tool to express these results. Intuitively, it offers a notion of many-one reducibility for partial multi-valued functions $f :\subseteq X \rightrightarrows Y$ that can be used to compare computational problems in a very natural and straightforward way.

Work on probabilistic notions in this setting has been started by Brattka and Pauly [15], Dorais, Dzhafarov, Hirst, Mileti and Shafer [18], Bienvenu and Porter [2] and others. In recent work Brattka, Gherardi and Hölzl [7, 8] have proposed the concept of Las Vegas computable functions and studied its computational power. The problem of computing Nash equilibria is an example of a Las Vegas computable problem, which is not deterministically computable.

Roughly speaking, a function $f :\subseteq X \rightrightarrows Y$ is *Las Vegas computable* if it can be computed on a Turing machine upon input of (a name of) some $x \in \mathrm{dom}(f)$ with the help of an additional advice $r \in 2^{\mathbb{N}}$ subject to the following conditions:

1. If the advice $r \in 2^{\mathbb{N}}$ is not helpful, then the machine recognizes this in finite time and stops the computation with a failure signal.
2. If the advice $r \in 2^{\mathbb{N}}$ is helpful, then the machine computes forever and produces a correct result, that is, (a name of) some $y \in f(x)$.
3. The set of helpful advices r for each fixed name of input x has to be of positive measure.

We continue this work in the present article and propose the new concept of Monte Carlo computability, which is different from Las Vegas computability in that we relax condition 1. Roughly speaking, we consider $f :\subseteq X \rightrightarrows Y$ as *Monte Carlo computable* if the fact that the advice $r \in 2^{\mathbb{N}}$ is not helpful can be recognized (only) in the limit. As a consequence of this relaxation such a machine might compute forever without producing a correct result. However, with positive probability it will produce a correct result.

In the Weihrauch lattice we can identify the classes $\mathsf{WKL}, \mathsf{WWKL}$ and $\mathsf{WWKL}' \times \mathsf{C}'_{\mathbb{N}}$ as being complete for non-deterministically computable, Las Vegas computable and Monte Carlo computable problems, respectively. All these classes are variants of Weak Kőnig's Lemma (WKL) and will be formally defined below. In a vague analogy these classes correspond to the complexity classes $\mathsf{NP}, \mathsf{ZPP}$ and BPP, respectively.

In contrast to the situation in computational complexity theory, we can separate the classes $\mathsf{WKL}, \mathsf{WWKL}$ and $\mathsf{WWKL}' \times \mathsf{C}'_{\mathbb{N}}$ and provide natural computational problems as witnesses for these separations. The picture that emerges is illustrated in Figure 1. The problems given in the picture have been studied before:

1. Differentiation is the problem $d\colon\subseteq \mathcal{C}[0,1] \to \mathcal{C}[0,1], f \mapsto f'$ to determine the derivative of a continuously differentiable function $f\colon [0,1] \to \mathbb{R}$ [27] and it is neither non-deterministically computable nor probabilistically computable in any form [8].
2. The fixed point problem $\mathsf{BFT}\colon \mathcal{C}([0,1]^n, [0,1]^n) \rightrightarrows [0,1]^n, f \mapsto \{x\colon f(x) = x\}$, that is, the problem of determining a fixed point of a continuous function $f\colon [0,1]^n \to [0,1]^n$ for $n \geq 2$ [14], is non-deterministically computable but not probabilistically computable in any form [8]. BFT stands for "Brouwer Fixed Point Theorem".
3. The zero problem $\mathsf{IVT}\colon\subseteq \mathcal{C}[0,1] \rightrightarrows [0,1], f \mapsto f^{-1}\{0\}$, that is, the problem that maps every continuous function $f\colon [0,1] \to \mathbb{R}$ with $f(0) \cdot f(1) < 0$ to one of its zeros [5], is non-deterministically computable, has a Monte Carlo algorithm, but is not Las Vegas computable [8]. IVT stands for "Intermediate Value Theorem".
4. Nash is the problem that maps a bi-matrix game $(A, B) \in \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n}$ to one of its Nash equilibria [23] and is Las Vegas computable [8].
5. Sorting stands for the problem $\mathsf{SORT}_2\colon 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ of sorting a binary sequence [22]. It is not non-deterministically computable but has a Monte Carlo algorithm, as we will show.

The given classifications of these problems (also illustrated in Figure 1) follow from results in the given references, except for the case of sorting, which we will precisely define and discuss in Section 5. This section contains the main technical contributions of this article. In Section 2 we start with recalling the definition of Weihrauch reducibility and some basic algebraic operations, followed by an introduction of the concept of Monte Carlo computability in Section 3. In Section 4 we discuss versions of Weak Weak Kőnig's Lemma. We close this article with a brief discussion of algebraic computation models in Section 6.

2 The Weihrauch Lattice

Formally, the Weihrauch lattice is formed by equivalence classes of partial multi-valued functions $f\colon\subseteq X \rightrightarrows Y$ on represented spaces X, Y. We will simply call such functions *problems* here and they are, in fact, computational challenges in the sense that for every $x \in \mathrm{dom}(f)$ the goal is to find *some* $y \in f(x)$. In this case $\mathrm{dom}(f)$ contains the admissible instances x of the problem and for each instance x the set $f(x)$ contains the corresponding solutions. Some typical problems f such as solving some type of equation or sorting a given sequence are mentioned above.

A *represented space* (X, δ) is a set X together with a surjective partial map $\delta\colon\subseteq \mathbb{N}^{\mathbb{N}} \to X$ that assigns *names* $p \in \mathbb{N}^{\mathbb{N}}$ to points $\delta(p) = x \in X$. These representations allow us to describe computations on all representable spaces using Turing machines that operate on the names corresponding to points in the space. We refer the reader to [28, 13] for details.

For problems $f\colon\subseteq X \rightrightarrows Y$ and $g\colon\subseteq Y \rightrightarrows Z$ we define the composition $g \circ f\colon\subseteq X \rightrightarrows Z$ by $g \circ f(x) = \{z \in Z\colon (\exists y \in f(x))\ z \in g(y)\}$, where $\mathrm{dom}(g \circ f) := \{x \in X\colon f(x) \subseteq \mathrm{dom}(g)\}$. We also denote the composition briefly by gf.

The intuition behind Weihrauch reducibility is that $f \leq_{\mathrm{W}} g$ holds if there is a computational procedure for solving f during which a single application of the computational resource g is allowed. There are actually two slightly different formal versions of this reduction, which are both needed. In expressions like $H(x, gK(x))$ we tacitly use the definition of the composition as given above.

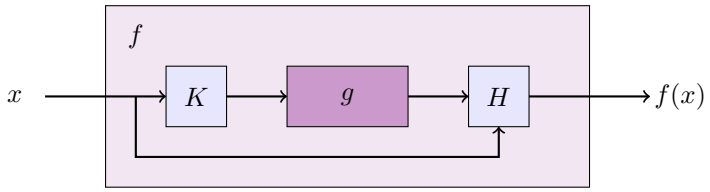

Figure 2 Visualization of Weihrauch reducibility $f \leq_{\mathrm{W}} g$.

▶ **Definition 1** (Weihrauch reducibility)**.** Let $f\colon\subseteq X \rightrightarrows Y$ and $g\colon\subseteq W \rightrightarrows Z$ be problems.

1. f is called *Weihrauch reducible* to g, in symbols $f \leq_{\mathrm{W}} g$, if there are computable $K\colon\subseteq X \rightrightarrows W$, $H\colon\subseteq X \times Z \rightrightarrows Y$ such that $\emptyset \neq H(x, gK(x)) \subseteq f(x)$ for all $x \in \mathrm{dom}(f)$.
2. f is called *strongly Weihrauch reducible* to g, in symbols $f \leq_{\mathrm{sW}} g$, if there are computable $K\colon\subseteq X \rightrightarrows W$, $H\colon\subseteq Z \rightrightarrows Y$ such that $\emptyset \neq HgK(x) \subseteq f(x)$ for all $x \in \mathrm{dom}(f)$.

The concept of Weihrauch reducibility is illustrated in Figure 2. The strong version of Weihrauch reducibility can be illustrated similarly without the direct input access of H.

Weihrauch reducibility induces a lattice with a rich and very natural algebraic structure. We briefly summarize some of these algebraic operations for problems $f\colon\subseteq X \rightrightarrows Y$ and $g\colon\subseteq W \rightrightarrows Z$:

- $f \times g$ is the *product* of f and g and represents the parallel evaluation of problem f on some input x and g on some input w.
- $f * g := \sup\{f_0 \circ g_0\colon f_0 \leq_{\mathrm{W}} f \text{ and } g_0 \leq_{\mathrm{W}} g\}$ is the *compositional product* and represents the consecutive usage of the problem f after the problem g.
- $f^* := \bigsqcup_{n=0}^{\infty} f^n$ is the *finite parallelization* and allows an evaluation of the n–fold product f^n for some arbitrary given $n \in \mathbb{N}$.
- f' denotes the *jump* of f, which is formally the same problem, but the input representation δ_X of X is replaced by its jump $\delta'_X := \delta_X \circ \lim$.

Here $\lim\colon\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}, \langle p_0, p_1, p_2, ...\rangle \mapsto \lim_{i\to\infty} p_i$ is the usual limit map on Baire space for $p_i \in \mathbb{N}^{\mathbb{N}}$, where $\langle\ \rangle$ denotes a standard infinite tupling function. One can also define a coproduct operation $\sqcup$ and a sum operation $\sqcap$ that play the role of supremum and infimum for ordinary Weihrauch reducibility $\leq_{\mathrm{W}}$, respectively. But we are not going to use these operations here. The resulting Weihrauch lattice is not complete as infinite suprema do not need to exist, but the supremum $f * g$ always exists as shown by Brattka and Pauly [16]. The finite parallelization is a closure operator in the Weihrauch lattice. Further information on the algebraic structure can be found in [16].

An important problem in the Weihrauch lattice is *closed choice* $\mathsf{C}_X\colon\subseteq \mathcal{A}_-(X) \rightrightarrows X$, $A \mapsto A$, which maps every closed set $A \subseteq X$ to its points. The crucial fact here is that closed sets $A \in \mathcal{A}_-(X)$ are represented with respect to negative information, essentially by enumerating open balls that exhaust their complement. That is, closed choice C_X is the following problem: given a closed set A by a description that lists everything that does not belong to A, find a point $x \in A$ (see [4] for further information). We also consider PC_X, which is the restriction of C_X to sets of positive measure, where we assume that we have some given natural Borel measure on X. In case of Cantor space $X = 2^{\mathbb{N}}$ we are going to use the uniform measure $\mu_{2^{\mathbb{N}}}$, in case of the reals $X = \mathbb{R}$ we use the Lebesgue measure $\mu_{\mathbb{R}}$. Different classes of problems have been characterized by different forms of closed choice in the following ways [4, 8]:

- $f \leq_{\mathrm{W}} \mathsf{C}_{\mathbb{N}} \iff f$ is computable with finitely many mind changes.
- $f \leq_{\mathrm{W}} \mathsf{C}_{2^{\mathbb{N}}} \iff f$ is non-deterministically computable.
- $f \leq_{\mathrm{W}} \mathsf{PC}_{2^{\mathbb{N}}} \iff f$ is Las Vegas computable.

Here non-deterministic computability is understood in the way defined by Martin Ziegler [29] for the advice space $2^{\mathbb{N}}$.

3 Monte Carlo Computability

The notion of Las Vegas computability was introduced by Brattka, Gherardi and Hölzl [7, 8] and analogously we are going to introduce the notion of Monte Carlo computability here. The intuition of Monte Carlo computability was already described in the introduction and it is illustrated in Figure 3. The essential difference between Las Vegas and Monte Carlo computations is that in the former case we can recognize the failure of the advice in finite time, whereas in the latter case we can only recognize the failure in the limit. Indeed, one can even relax this condition further and obtain a whole hierarchy of notions of Monte Carlo computations, but we are not going to discuss this hierarchy here.

Instead of introducing Monte Carlo machines formally, we implicitly define them by using ordinary computable functions. Essentially, the computation of a Monte Carlo machine is governed by two functions F_1 and F_2. Roughly speaking, F_1 is responsible to provide the result of the computation upon some input and some additional advice $r \in 2^{\mathbb{N}}$ and F_2 is responsible to recognize whether the advice r fails.

The status of the advice is captured using Sierpiński space $\mathbb{S} = \{0, 1\}$, which is equipped with the topology $\{\emptyset, \{1\}, \mathbb{S}\}$ and a corresponding representation $\delta_{\mathbb{S}}$. This space provides an asymmetric way to capture the status of the computation, very much in the same way as being computably enumerable is an asymmetric version of decidability. The point is that failure of an advice is supposed to be a recognizable event (in the limit), but success cannot necessarily be recognized in the same way. Monte Carlo computability can now be formalized as follows.

▶ **Definition 2** (Monte Carlo computability)**.** Let (X, δ_X) and (Y, δ_Y) be represented spaces. A problem $f\colon\subseteq X \rightrightarrows Y$ is said to be *Monte Carlo computable* if there exists a computable function $F_1\colon\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ and a limit computable function $F_2\colon\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{S}$ such that $\langle \mathrm{dom}(f\delta_X) \times 2^{\mathbb{N}} \rangle \subseteq \mathrm{dom}(F_2)$ and for each $p \in \mathrm{dom}(f\delta_X)$ the following hold:

1. $S_p := \{r \in 2^{\mathbb{N}} \colon F_2\langle p, r\rangle = 0\}$ is non-empty and $\mu_{2^{\mathbb{N}}}(S_p) > 0$,
2. $\delta_Y F_1\langle p, r\rangle \in f\delta_X(p)$ for all $r \in S_p$.

This difference to Las Vegas computability lies in the fact that F_2 is only required to be limit computable (which is the same as effectively $\mathbf{\Sigma}^0_2$–measurable) and not necessarily computable. While computable characteristic functions $\chi_{2^{\mathbb{N}}\setminus A}\colon 2^{\mathbb{N}} \to \mathbb{S}$ capture exactly co-c.e. closed sets A (that is, effective $\mathbf{\Pi}^0_1$–sets in the Borel hierarchy), limit computable characteristic functions $\chi_{2^{\mathbb{N}}\setminus A}$ capture exactly effective G_δ–sets A (that is, effective $\mathbf{\Pi}^0_2$–sets in the Borel hierarchy) (see Pauly [25]). Hence, it should not come as a big surprise that Monte Carlo computability is closely related to choice for G_δ–sets, which we define next. By $\mathbf{\Pi}^0_2(2^{\mathbb{N}})$ we denote the class of G_δ–subsets of $2^{\mathbb{N}}$.

▶ **Definition 3** (Positive G_δ–choice)**.** By $\mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}}\colon\subseteq \mathbf{\Pi}^0_2(2^{\mathbb{N}}) \rightrightarrows 2^{\mathbb{N}}, A \mapsto A$ we denote the *positive G_δ–choice problem*, defined for all $A \in \mathbf{\Pi}^0_2(2^{\mathbb{N}})$ with $\mu_{2^{\mathbb{N}}}(A) > 0$.

The crucial idea for a simple proof of the following result is to use a synthetic representation for the class of G_δ–sets. There is a canonical function space representation $[\delta_{2^{\mathbb{N}}} \to \delta_{\mathbb{S}}]$ of the continuous functions $\chi\colon 2^{\mathbb{N}} \to \mathbb{S}$ and likewise $[\delta_{2^{\mathbb{N}}} \to \delta'_{\mathbb{S}}]$ is a representation of the $\mathbf{\Sigma}^0_2$–measurable functions [25] that we are going to use to represent $\mathbf{\Pi}^0_2(2^{\mathbb{N}})$. This synthetic representation enables us to apply the methods of evaluation and type conversion. Hence we

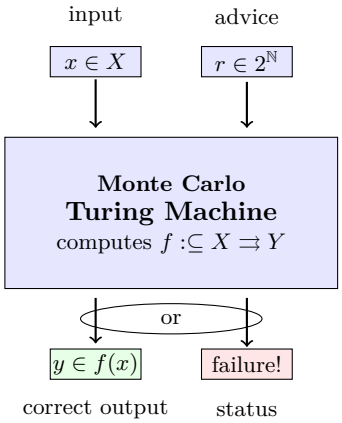

Figure 3 Illustration of a Monte Carlo machine that computes $f\colon\subseteq X \rightrightarrows Y$.

can transfer the proof of [4, Theorem 7.2] by Brattka, de Brecht and Pauly literally to our setting.

▶ **Theorem 4** (Monte Carlo computability). *$f \leq_{\mathrm{W}} \mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}}$ if and only if f is Monte Carlo computable.*

The class $\mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}}$ has been studied before and the following theorem was proved by Brattka, Gherardi, Hölzl, Nobrega and Pauly [9].

▶ **Theorem 5** (Positive G_δ–Choice). $\mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}} \equiv_{\mathrm{sW}} \mathsf{PC}'_{\mathbb{R}}$.

In our context $\mathsf{PC}'_{\mathbb{R}}$ is somewhat easier to handle than $\mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}}$ and Theorems 4 and 5 lead to the following corollary.

▶ **Corollary 6** (Monte Carlo computability). *$f \leq_{\mathrm{W}} \mathsf{PC}'_{\mathbb{R}}$ if and only if f is Monte Carlo computable.*

Bienvenu and Kuyper [1] answered a number of questions related to composition and they proved the following result on the compositional product of $\mathsf{PC}'_{\mathbb{R}}$.

▶ **Theorem 7** (Composition). $\mathsf{PC}'_{\mathbb{R}} * \mathsf{PC}'_{\mathbb{R}} \equiv_{\mathrm{W}} \mathsf{PC}'_{\mathbb{R}}$.

In light of Theorem 4 and Theorem 5 we obtain a second independent proof of this theorem along the lines of the Independent Choice Theorem of Brattka, Gherardi and Hölzl [8, Theorem 4.3], which is essentially based on Fubini's Theorem. The synthetic representation of $\mathbf{\Pi}^0_2(2^{\mathbb{N}})$ allows us to transfer the proof of the Independent Choice Theorem directly to a proof of the fact that $\mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}} * \mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}} \equiv_{\mathrm{W}} \mathbf{\Pi}^0_2\mathsf{PC}_{2^{\mathbb{N}}}$ and hence we obtain Theorem 7 with the help of Theorem 5. The importance of Theorem 7 for us lies in the following conclusion.

▶ **Corollary 8.** *Monte Carlo computable functions are closed under composition.*

This conclusion ensures that Monte Carlo computability satisfies one of the necessary conditions that any reasonable concept of computability should satisfy. Due to the definition it is also clear that every Las Vegas computable function is Monte Carlo computable.

▶ **Corollary 9.** *Every Las Vegas computable function is Monte Carlo computable.*

The inverse implication is clearly false, as there are functions that are Monte Carlo computable but not Las Vegas computable. An example is the equality test $=: \mathbb{R}\times\mathbb{R} \to \{0,1\}$; it is Monte Carlo computable since it is reducible to $\mathsf{C}_{\mathbb{N}} \leq_{\mathrm{W}} \mathsf{PC}'_{\mathbb{R}}$, but cannot be Las Vegas computable since it is not even non-deterministically computable [6].

4 Weak Weak Kőnig's Lemma and Jumps

In this section we discuss the relation of Weak Weak Kőnig's Lemma to Monte Carlo computations. Weak Kőnig's Lemma and Weak Weak Kőnig's Lemma are principles that have been intensively studied in reverse mathematics [26]. The classical lemma of Kőnig says (in its weak version) that every infinite binary tree has an infinite path. Here we understand Weak Kőnig's Lemma as the mathematical problem

$$\mathsf{WKL}\colon\subseteq \mathrm{Tr} \rightrightarrows 2^{\mathbb{N}}, T \mapsto [T]$$

that maps an infinite binary tree $T \subseteq 2^*$ to an infinite path $p \in [T]$ of this tree. By Tr we denote the set of all binary trees (represented via their characteristic functions) and by $[T]$ we denote the set of infinite paths of such a tree. We assume that $\mathrm{dom}(\mathsf{WKL})$ is the set of infinite binary trees. Weak Weak Kőnig's Lemma is the restriction of WKL to trees T such that $\mu_{2^{\mathbb{N}}}([T]) > 0$, that is, such that the set of infinite paths has positive measure. Some basic facts known about these principles are the following (see [21, 8, 10]):

- $\mathsf{WKL} \equiv_{\mathrm{sW}} \mathsf{C}_{2^{\mathbb{N}}}$.
- $\mathsf{WWKL} \equiv_{\mathrm{sW}} \mathsf{PC}_{2^{\mathbb{N}}}$.
- $\mathsf{WWKL} \times \mathsf{C}_{\mathbb{N}} \equiv_{\mathrm{sW}} \mathsf{PC}_{\mathbb{R}}$.
- $\mathsf{WWKL} <_{\mathrm{sW}} \mathsf{WKL}$ and $\mathsf{WWKL} <_{\mathrm{sW}} \mathsf{PC}_{\mathbb{R}}$.
- $\mathsf{K}_{\mathbb{N}} <_{\mathrm{sW}} \mathsf{WWKL}$ and $\mathsf{C}_{\mathbb{N}} <_{\mathrm{sW}} \mathsf{PC}_{\mathbb{R}}$.

Here $\mathsf{K}_{\mathbb{N}} := \mathsf{C}_2^*$ is also called *compact choice* because it can be seen as $\mathsf{C}_{\mathbb{N}}$ restricted to sets $A \subseteq \mathbb{N}$ that are given together with an upper bound.

In the context of Monte Carlo computability we need to transfer some of these results to jumps of the involved principles. For the forwards direction one can usually use monotonicity of jumps with respect to strong Weihrauch reducibility. The following lemma was proved by Brattka, Gherardi and Marcone [11, Proposition 5.6].

▶ **Lemma 10** (Monotonicity). $f \leq_{\mathrm{sW}} g \Longrightarrow f' \leq_{\mathrm{sW}} g'$.

We note that a corresponding result for ordinary Weihrauch reducibility $\leq_{\mathrm{W}}$ does not hold. This is one of the reasons why the study of strong Weihrauch reductions seems to be unavoidable. Even if one is interested only in results about ordinary Weihrauch reducibility of jumps, this typically requires to study strong Weihrauch reductions.

Surprisingly, we were also able to prove a certain inverse result of the above monotonicity property, which we formulate next. Here "relative to the halting problem" is supposed to mean that the reduction functions H, K used for the reduction $f \leq_{\mathrm{W}} g$ both have access to the halting problem.

▶ **Theorem 11** (Jumps and relativization). $f' \leq_{\mathrm{W}} g' \Longrightarrow f \leq_{\mathrm{W}} g$ *relative to the halting problem.*

Proof. Let $f' \leq_{\mathrm{W}} g'$. Then there are computable functions $H, K\colon\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ such that $H\langle r, GK(r)\rangle$ is a name for an output of f' on the input specified by $r \in \mathbb{N}^{\mathbb{N}}$, whenever G is a realizer for g'. For r actually any sequence is allowed that converges to an input q of f and $K(r)$ must be a sequence converging to a name of an input of g in this situation. Without loss of generality we can assume that $\mathrm{range}(K) \subseteq 2^{\mathbb{N}}$. By a theorem of Brattka, Hendtlass

and Kreuzer [12, Theorem 14.11] there are functions $K_0 :\subseteq \mathbb{N}^{\mathbb{N}} \to 2^{\mathbb{N}}$ and $K_1 :\subseteq \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ that are computable with access to the halting problem and such that

$$K_0(q) = \lim KK_1(q) \text{ and } \lim K_1(q) = q$$

for all $q \in \mathrm{dom}(\lim \circ K \circ \lim^{-1})$. Then $K_1(q) = r$ is a sequence that converges to q and such that $K_0(q) = \lim K(r)$ is a name of an input of g. Let H_0 be defined by $H_0\langle q, u\rangle := H\langle K_1(q), u\rangle$ for all q, u. Then H_0 is computable relative to the halting problem and we obtain that $G \lim$ is a realizer of g' for every realizer G of g and hence

$$H_0\langle q, GK_0(q)\rangle = H\langle K_1(q), GK_0(q)\rangle = H\langle r, G \lim K(r)\rangle,$$

which is name for an output of f' on input r and hence an output for f on input $\lim r = q$. This proves $f \leq_{\mathrm{W}} g$ relative to the halting problem. ◀

An analogous statement holds for $\leq_{\mathrm{sW}}$ in place of $\leq_{\mathrm{W}}$. Often separations of problems are proved in a topological way by showing that there are not even continuous reduction functions H, K. In such a situation, H, K cannot even be computable with respect to the halting problem. Hence, topological separations of f and g are very useful when it comes to separating f' and g'.

The diagram in the upper half of Figure 5 illustrates some important reductions for jumps of Weak Weak Kőnig's Lemma and related principles

5 Sorting

Sorting infinite sequences is a basic computational task that was introduced and studied by Neumann and Pauly [22] in the binary case. We generalize this problem by defining $\mathsf{SORT}_n : \{0, 1, ..., n-1\}^{\mathbb{N}} \to \{0, 1, ..., n-1\}^{\mathbb{N}}$ by

$$\mathsf{SORT}_n(p) := 0^{k_0}1^{k_1}...(m-1)^{k_{m-1}}\widehat{m}$$

if $m < n$ is the smallest digit that appears infinitely often in p and each digit $i < m$ appears exactly k_i times in p. Here $\widehat{m} = mmm...$ denotes the infinite sequence which has the constant value m. This definition is understood such that $\mathsf{SORT}_n(p) = \widehat{0}$ if 0 appears infinitely often in p. In Figure 4 SORT_5 is illustrated.

For $n \leq 1$ the problem SORT_n is computable, since SORT_0 is the nowhere defined function and SORT_1 is the constant function. Neumann and Pauly [22, Proposition] proved that $\mathsf{C}_{\mathbb{N}} \leq_{\mathrm{W}} \mathsf{SORT}_2$ and in fact it is easy to see that even a strong reduction holds.

▶ **Proposition 12.** $\mathsf{C}_{\mathbb{N}} \leq_{\mathrm{sW}} \mathsf{SORT}_2$.

Here we want to discuss probabilistic solutions for the problem SORT_n and the interesting observation is the following.

▶ **Proposition 13.** $\mathsf{SORT}_n \leq_{\mathrm{sW}} \mathsf{WWKL}'$ *for all* $n \in \mathbb{N}$.

Proof. Given a sequence $p \in \{0, ..., n-1\}^{\mathbb{N}}$ as input to SORT_n we want to compute $\mathsf{SORT}_n(p)$ with the help of WWKL'. We produce a sequence of binary trees $(T_i)_i$ with $T_i := \bigcup_{m=0}^{n-1} T_{i,m}$. Here for each $m \in \{0, ..., n-1\}$

$$T_{i,m} := 0^m 10^{k_0} 10^{k_1} ... 10^{k_{m-1}} 1\{0, 1\}^{k_m},$$

where k_j denotes the number of appearances of digit $j \in \{0, ..., m\}$ in the initial segment $p(0), ..., p(i)$. The sequence $(T_{i,m})_i$ converges to a tree S_m. If no $j \leq m$ appears infinitely

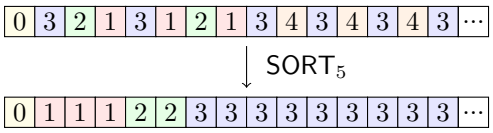

Figure 4 Application of SORT_5 to some example sequence (that continues alternately with $4, 3$).

often in p, then $[S_m] = \emptyset$. If m appears infinitely often in p and no smaller $j < m$ appears infinitely often in p, then

$$[S_m] = 0^m 10^{k_0} 10^{k_1} ... 10^{k_{m-1}} 1\{0,1\}^{\mathbb{N}},$$

where k_j is the number of appearances of digit j in p for $j \in \{0, ..., m-1\}$. If $l < m$ is the minimal digit that appears infinitely often in p, then

$$[S_m] = \{0^m 10^{k_0} 10^{k_1} ... 10^{k_{l-1}} \widehat{10}\},$$

where k_j is the number of appearances of digit j in p for $j \in \{0, ..., l-1\}$. The sequence $(T_i)_i$ converges to the tree $T := \bigcup_{m=0}^{n-1} S_m$. Since one digit $m < n$ is the minimal digit that appears infinitely often in p, we have $\mu([T]) \geq \mu([S_m]) > 0$. Given some $q \in [T]$, we can reconstruct $\mathsf{SORT}_n(p)$, since there is some $m < n$ and there is some $l < m$ or some $r \in \{0,1\}^{\mathbb{N}}$ such that exactly one of the following cases holds:

1. $q = 0^m 10^{k_0} 10^{k_1} ... 10^{k_{l-1}} \widehat{10} \implies \mathsf{SORT}_n(p) = 0^{k_0} 1^{k_1} ... (l-1)^{k_{l-1}} \widehat{l}$,
2. $q = 0^m 10^{k_0} 10^{k_1} ... 10^{k_{m-1}} 1r \implies \mathsf{SORT}_n(p) = 0^{k_0} 1^{k_1} ... (m-1)^{k_{m-1}} \widehat{m}$. ◀

Brattka, Gherardi and Hölzl proved that WWKL (and hence WWKL') is strongly idempotent [8, Corollary 4.5], that is, $\mathsf{WWKL}' \times \mathsf{WWKL}' \equiv_{\mathrm{sW}} \mathsf{WWKL}'$, and hence Proposition 13 can be strengthened in the following way.

▶ **Corollary 14.** $\mathsf{SORT}_n^* \leq_{\mathrm{sW}} \mathsf{WWKL}'$ *for all* $n \in \mathbb{N}$.

Since $\mathsf{WWKL}' \leq_{\mathrm{W}} \mathsf{PC}'_{\mathbb{R}}$ we also obtain the following conclusion.

▶ **Corollary 15.** SORT_n^* *is Monte Carlo computable for every* $n \in \mathbb{N}$.

The next observation is that the Intermediate Value Theorem IVT is reducible to SORT_2. We study IVT here in the slightly easier form of connected choice $\mathsf{CC}_{[0,1]}$, that is, $\mathsf{C}_{[0,1]}$ restricted to connected subsets of $[0,1]$. The equivalence $\mathsf{IVT} \equiv_{\mathrm{sW}} \mathsf{CC}_{[0,1]}$ was proved by Brattka and Gherardi [5, Theorem 6.2] (where $\mathsf{CC}_{[0,1]}$ appears under the name C_{I}).

▶ **Proposition 16.** $\mathsf{CC}_{[0,1]} \leq_{\mathrm{W}} \mathsf{SORT}_2$.

Proof. Let two monotone rational sequences $(a_n)_n$ and $(b_n)_n$ in $[0,1]$ be given, the first one increasing, the second one decreasing, and such that $a_n \leq b_n$ for all n. We computably produce an input p for SORT_2 by writing a 1 every second step and by producing occasionally a 0 according to the following algorithm: whenever we find an $n \in \mathbb{N}$ such that $|b_n - a_n| < 2^{-k}$, then we ensure that we have k digits 0 included in p. Hence, p will include exactly k zeros if k is the largest number such that there is some n with $|b_n - a_n| < 2^{-k}$ and it will include infinitely many zeros if for every k there is such an n. Given the output of $\mathsf{SORT}_2(p)$ and the original input $(a_n)_n$ and $(b_n)_n$ we can find a point $x \in [0,1]$ with $a_n \leq x \leq b_n$ for all n as follows. If we see at least k zeros in $\mathsf{SORT}_2(p)$, then we search for n with $|b_n - a_n| < 2^{-k}$ and produce $x_k := a_n + \frac{1}{2}(b_n - a_n)$ as approximation of x of precision 2^{-k}. In the moment where

we see that there are k and not more zeros in $\mathsf{SORT}_2(p)$, that is, we see the first 1 at position $k+1$, we continue to produce $x_{k+i} := x_k$ as approximation for x of any higher precision 2^{-k-i}. Since there is no n with $|b_n - a_n| < 2^{-k-1}$, it is guaranteed that $a_n \leq x_{k+i} \leq b_n$ for all $i \in \mathbb{N}$. ◀

In particular, this result implies that zero finding, that is, the Intermediate Value Theorem IVT, is Monte Carlo computable.

▶ **Corollary 17.** *The Intermediate Value Theorem* IVT *is Monte Carlo computable.*

Brattka, Gherardi and Hölzl proved $\mathsf{CC}_{[0,1]} \leq_{\mathrm{W}} \mathsf{WWKL}'$ [8, Corollary 15.9], which also implies Corollary 17. However, Proposition 16 generalizes the aforementioned result via Proposition 13. We note that $\mathsf{CC}_{[0,1]} \not\leq_{\mathrm{sW}} \mathsf{SORT}_2$ since $\mathsf{IVT} \not\leq_{\mathrm{sW}} \mathsf{WWKL}'$ by [8, Corollary 15.8]. Brattka and Rakotoniaina proved $\mathsf{CC}_{[0,1]} \leq_{\mathrm{W}} \mathsf{C}'_{\mathbb{N}}$ [17, Proposition 5.21], a result which is generalized by the following observation.

▶ **Proposition 18.** $\mathsf{SORT}_n \leq_{\mathrm{sW}} \mathsf{C}'_{\mathbb{N}}$ *for all* $n \in \mathbb{N}$.

Proof. By [11, Theorem 9.4] we have $\mathsf{CL}_{\mathbb{N}} \equiv_{\mathrm{sW}} \mathsf{C}'_{\mathbb{N}}$, where $\mathsf{CL}_{\mathbb{N}}$ denotes the cluster point problem (i.e., the problem to find a cluster point of a given sequence of natural numbers that has one). Given a sequence $p \in \{0, 1, ..., n-1\}^{\mathbb{N}}$ we compute a sequence $q \in \mathbb{N}^{\mathbb{N}}$ as follows: we inspect p and whenever we find a new occurrence of the digit $m \in \{0, 1, ..., n-1\}$, then we add a number $\langle m, \langle k_0, k_1, ..., k_{m-1}\rangle\rangle \in \mathbb{N}$ to q, where each $k_i \in \mathbb{N}$ counts the occurrences of digit i in p so far. One fixed such tuple $\langle m, \langle k_0, k_1, ..., k_{m-1}\rangle\rangle$ appears infinitely often in q if and only if m appears infinitely often in p and each number $i \in \{0, 1, ..., m-1\}$ appears exactly k_i times in p altogether. $\mathsf{CL}_{\mathbb{N}}(q)$ determines one such tuple that appears infinitely often and hence it is easy to reconstruct $\mathsf{SORT}_n(p)$ from $\mathsf{CL}_{\mathbb{N}}(q)$. ◀

We note that the proof even shows $\mathsf{SORT}_n \leq_{\mathrm{sW}} \mathsf{UC}'_{\mathbb{N}}$, where $\mathsf{UC}'_{\mathbb{N}}$ is the unique version of $\mathsf{C}'_{\mathbb{N}}$. Proposition 18 also yields an independent proof of Corollary 15. With the help of the fact that $\mathsf{C}'_{\mathbb{N}} \times \mathsf{C}'_{\mathbb{N}} \equiv_{\mathrm{sW}} \mathsf{C}'_{\mathbb{N}}$ we obtain the following corollary.

▶ **Corollary 19.** $\mathsf{SORT}^*_n \leq_{\mathrm{sW}} \mathsf{C}'_{\mathbb{N}}$ *for all* $n \in \mathbb{N}$.

One could ask whether this result can be strengthened to the upper bound $\mathsf{K}'_{\mathbb{N}}$, which is not the case.

▶ **Proposition 20.** $\mathsf{CC}_{[0,1]} \not\leq_{\mathrm{W}} \mathsf{K}'_{\mathbb{N}}$.

Proof. By [11, Theorem 9.4] we have $\mathsf{K}'_{\mathbb{N}} \equiv_{\mathrm{sW}} \mathsf{BWT}_{\mathbb{N}}$, where $\mathsf{BWT}_{\mathbb{N}}$ denotes the restriction of $\mathsf{CL}_{\mathbb{N}}$ to bounded sequences. Let us assume that $\mathsf{CC}_{[0,1]} \leq_{\mathrm{W}} \mathsf{BWT}_{\mathbb{N}}$ holds via computable reduction functions K, H. We construct sequences $(a_n)_n$ and $(b_n)_n$ of rational numbers in $[0,1]$ with $a_n \leq a_{n+1} \leq b_{n+1} \leq b_n$ for all $n \in \mathbb{N}$ as an input for $\mathsf{CC}_{[0,1]}$. We start with choosing $a_n := 0$ and $b_n := 1$ for all $n \in \mathbb{N}$. Upon this input K will produce a sequence $(x_n)_n$ of natural numbers as an input to $\mathsf{BWT}_{\mathbb{N}}$. This sequence has to contain only finitely many different numbers $y_0, ..., y_j$. Each of these numbers is a possible output of $\mathsf{BWT}_{\mathbb{N}}$ and hence for each of these numbers y_i and the original input, H has to select a point $z_i \in [0,1]$ with $a_n \leq z_i \leq b_n$ for all $n \in \mathbb{N}$. Since H is continuous, there is a certain input length k such that upon input of $a_0, ..., a_k$ and $b_0, ..., b_k$ the function H already approximates each number z_i up to precision 2^{-j-2}. It is clear that $j+1$ intervals of diameter 2^{-j-1} cannot cover the entire interval $[0,1]$ and in fact the uncovered part has non-empty interior. We can also assume that k is large enough such that by continuity of K actually all the numbers $y_0, ..., y_j$ already appear in the output of K upon input of $a_0, ..., a_k$ and $b_0, ..., b_k$. Now we

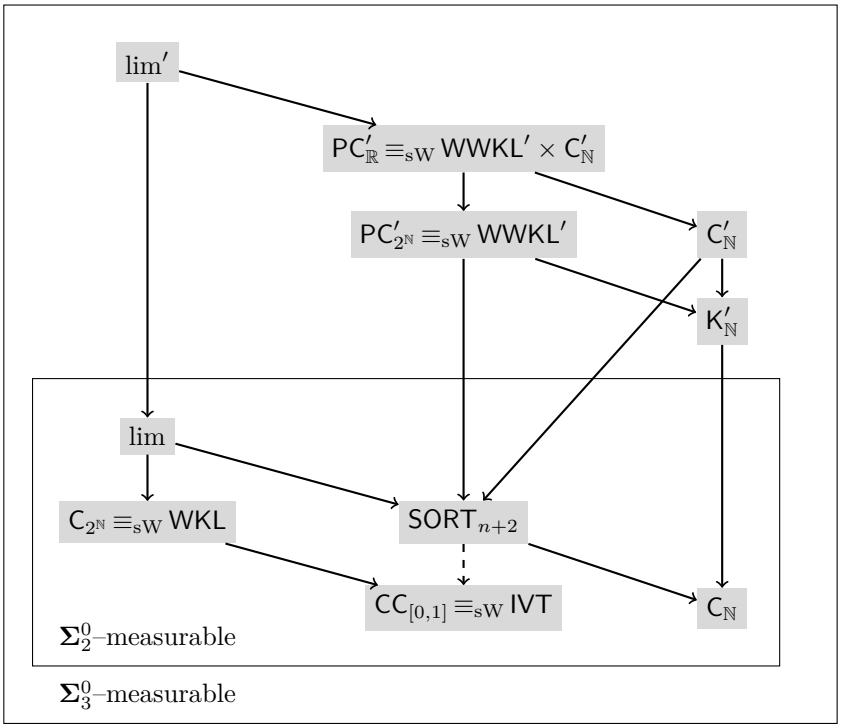

Figure 5 Sorting in the Weihrauch lattice.

can choose a new proper interval $[a, b]$ that does not overlap with any of the approximations of the z_i and we modify the original input by choosing $a_{k+i} := a$ and $b_{k+i} := b$ for all $i \in \mathbb{N}$. Upon this modified input K has to produce a new number $y \in \mathbb{N}$, which is not among the numbers $y_0, ..., y_j$, since otherwise the output that is produced by the reduction is not in $[a, b]$. We can now inductively repeat the above construction and in this way we construct an input $(a_n)_n$, $(b_n)_n$ to $\mathsf{CC}_{[0,1]}$ such that K produces a sequence $(x_n)_n$ with infinitely many different numbers x_n in its range (none of which appear infinitely often, in fact). This is not an admissible input to $\mathsf{BWT}_{\mathbb{N}}$ and hence a contradiction. ◀

Neumann and Pauly [22, Corollary 27] proved that SORT_2 is low_2 in the sense that $\lim * \lim * \mathsf{SORT}_2 \equiv_{\mathrm{W}} \lim * \lim$. However, it is not closed under composition and not low as, for instance, the following result shows.

▶ **Proposition 21.** $\mathsf{C}'_n \leq_{\mathrm{W}} \mathsf{SORT}_n * \mathsf{SORT}_n \leq_{\mathrm{W}} \mathsf{C}'_{\mathbb{N}}$ *for all* $n \in \mathbb{N}$.

Proof. We obtain $\mathsf{C}'_n \leq_{\mathrm{W}} \lim_n * \mathsf{SORT}_n \leq_{\mathrm{W}} \mathsf{C}_{\mathbb{N}} * \mathsf{SORT}_n \leq_{\mathrm{W}} \mathsf{SORT}_n * \mathsf{SORT}_n \leq_{\mathrm{W}} \mathsf{C}'_{\mathbb{N}}$. The first reduction holds since $\mathsf{SORT}_n(p)$ is always a converging sequence and its limit is a cluster point of p. We have $\mathsf{C}'_n \equiv_{\mathrm{W}} \mathsf{CL}_n$ for the cluster point problem CL_n by a result of Brattka, Gherardi and Marcone [11, Theorem 9.4]. The second reduction follows since $\lim_n \leq_{\mathrm{W}} \lim_{\mathbb{N}} \leq_{\mathrm{W}} \mathsf{C}_{\mathbb{N}}$ and the third reduction follows from Proposition 12. The reduction $\mathsf{SORT}_n * \mathsf{SORT}_n \leq_{\mathrm{W}} \mathsf{C}'_{\mathbb{N}}$ follows from Proposition 18 and the fact that $\mathsf{C}'_{\mathbb{N}} * \mathsf{C}'_{\mathbb{N}} \equiv_{\mathrm{W}} \mathsf{C}'_{\mathbb{N}}$ (which one can easily see by a direct proof). ◀

As a consequence of this result we obtain $\mathsf{SORT}_n \not\leq_{\mathrm{W}} \mathsf{WKL}$ for $n \geq 2$ since $\mathsf{WKL} * \mathsf{WKL} \equiv_{\mathrm{W}} \mathsf{WKL}$ by [21, Theorem 6.14] and hence we get the following conclusion.

▶ **Corollary 22.** SORT_n *is not non-deterministically computable for* $n \geq 2$.

In the diagram in Figure 5 we have collected some of the results on sorting. The solid lines indicate strong Weihrauch reductions against the direction of the arrows, the dashed line indicates an ordinary Weihrauch reduction against the direction of the arrow.

6 Algebraic Computation Models

One reason that Neumann and Pauly studied the binary sorting problem is that SORT_2^* characterizes the strength of strongly analytic machines as defined by Gärtner and Hotz [19, 20]. Essentially these strongly analytic machines are real random access machines over the field of real numbers (with computable constants) as studied by Blum, Shub and Smale and others [3] and extended by semantics that allows one to approximate the output. They are called *strongly* analytic if the machine also provides an error bound for the approximation of the output. Neumann and Pauly proved the following theorem [22, Observation 30, Corollary 34 and 11].

▶ **Theorem 23** (Algebraic machine models). *Consider a function of type $f\colon \mathbb{R}^* \to \mathbb{R}^*$.*

1. *If f is computable by a strongly analytic machine, then $f \leq_{\mathrm{W}} \mathsf{SORT}_2^*$ and SORT_2^* is equivalent to a function computable by a strongly analytic machine.*
2. *If f is computable by a BSS machine, then $f \leq_{\mathrm{W}} \mathsf{C}_{\mathbb{N}}$ and $\mathsf{C}_{\mathbb{N}}$ is equivalent to a function computable by a BSS machine.*

Using this result we obtain the following corollary; note that while the class of functions computable by strongly analytic machines is not closed under composition, the corollary *does* hold for such compositions as well.

▶ **Corollary 24.** *Any finite composition of functions $f\colon \mathbb{R}^* \to \mathbb{R}^*$ that are computable on strongly analytic machines is Monte Carlo computable.*

Since functions that are computable on a Blum, Shub and Smale machine (BSS machine) are a special case, we obtain the following corollary.

▶ **Corollary 25.** *Every function $f\colon \mathbb{R}^* \to \mathbb{R}^*$ that is computable on a BSS machine is Monte Carlo computable.*

Corollary 25 could already be deduced from the observation that $\mathsf{C}_{\mathbb{N}} \leq_{\mathrm{W}} \mathsf{PC}_{\mathbb{R}}$ and does not require our results on sorting. However, it is worth pointing out that in general functions computable on a BSS machine (even the equality test) are not non-deterministically computable and, in particular, not Las Vegas computable.

References

1 Laurent Bienvenu and Rutger Kuyper. Parallel and serial jumps of Weak Weak König's Lemma. In Adam Day, Michael Fellows, Noam Greenberg, Bakhadyr Khoussainov, Alexander Melnikov, and Frances Rosamond, editors, *Computability and Complexity: Essays Dedicated to Rodney G. Downey on the Occasion of His 60th Birthday*, volume 10010 of *Lecture Notes in Computer Science*, pages 201–217. Springer, Cham, 2017. doi:10.1007/978-3-319-50062-1_15.

2 Laurent Bienvenu and Christopher P. Porter. Deep Π^0_1 classes. *Bulletin of Symbolic Logic*, 22(2):249–286, 2016. doi:10.1017/bsl.2016.9.

3 Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and Real Computation*. Springer, New York, 1998.

4 Vasco Brattka, Matthew de Brecht, and Arno Pauly. Closed choice and a uniform low basis theorem. *Annals of Pure and Applied Logic*, 163:986–1008, 2012. doi:10.1016/j.apal.2011.12.020.

5 Vasco Brattka and Guido Gherardi. Effective choice and boundedness principles in computable analysis. *The Bulletin of Symbolic Logic*, 17(1):73–117, 2011. doi:10.2178/bsl/1294186663.

6 Vasco Brattka and Guido Gherardi. Weihrauch degrees, omniscience principles and weak computability. *The Journal of Symbolic Logic*, 76(1):143–176, 2011. doi:10.2178/jsl/1294170993.

7 Vasco Brattka, Guido Gherardi, and Rupert Hölzl. Las Vegas computability and algorithmic randomness. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science (STACS 2015)*, volume 30 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 130–142, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2015.130.

8 Vasco Brattka, Guido Gherardi, and Rupert Hölzl. Probabilistic computability and choice. *Information and Computation*, 242:249–286, 2015. doi:10.1016/j.ic.2015.03.005.

9 Vasco Brattka, Guido Gherardi, Rupert Hölzl, Hugo Nobrega, and Arno Pauly. Positive Borel choice. Unpublished draft, 2016.

10 Vasco Brattka, Guido Gherardi, Rupert Hölzl, and Arno Pauly. The Vitali covering theorem in the Weihrauch lattice. In Adam Day, Michael Fellows, Noam Greenberg, Bakhadyr Khoussainov, Alexander Melnikov, and Frances Rosamond, editors, *Computability and Complexity: Essays Dedicated to Rodney G. Downey on the Occasion of His 60th Birthday*, volume 10010 of *Lecture Notes in Computer Science*, pages 188–200. Springer, Cham, 2017. doi:10.1007/978-3-319-50062-1_14.

11 Vasco Brattka, Guido Gherardi, and Alberto Marcone. The Bolzano-Weierstrass theorem is the jump of weak Kőnig's lemma. *Annals of Pure and Applied Logic*, 163:623–655, 2012. doi:10.1016/j.apal.2011.10.006.

12 Vasco Brattka, Matthew Hendtlass, and Alexander P. Kreuzer. On the uniform computational content of computability theory. arXiv 1501.00433, 2015. arXiv:1501.00433.

13 Vasco Brattka, Peter Hertling, and Klaus Weihrauch. A tutorial on computable analysis. In S. Barry Cooper, Benedikt Löwe, and Andrea Sorbi, editors, *New Computational Paradigms: Changing Conceptions of What is Computable*, pages 425–491. Springer, New York, 2008. doi:10.1007/978-0-387-68546-5_18.

14 Vasco Brattka, Stéphane Le Roux, Joseph S. Miller, and Arno Pauly. The Brouwer fixed point theorem revisited. In Arnold Beckmann, Laurent Bienvenu, and Nataša Jonoska, editors, *Pursuit of the Universal*, volume 9709 of *Lecture Notes in Computer Science*, pages 58–67, Switzerland, 2016. Springer. 12th Conference on Computability in Europe, CiE 2016, Paris, France, June 27 - July 1, 2016. doi:10.1007/978-3-319-40189-8_6.

15 Vasco Brattka and Arno Pauly. Computation with advice. In Xizhong Zheng and Ning Zhong, editors, *CCA 2010, Proceedings of the Seventh International Conference on Computability and Complexity in Analysis*, Electronic Proceedings in Theoretical Computer Science, pages 41–55, 2010. doi:10.4204/EPTCS.24.9.

16 Vasco Brattka and Arno Pauly. On the algebraic structure of Weihrauch degrees. arXiv 1604.08348, 2016. arXiv:1604.08348.

17 Vasco Brattka and Tahina Rakotoniaina. On the uniform computational content of Ramsey's theorem. arXiv 1508.00471, 2015. arXiv:1508.00471.

18 François G. Dorais, Damir D. Dzhafarov, Jeffry L. Hirst, Joseph R. Mileti, and Paul Shafer. On uniform relationships between combinatorial problems. *Transactions of the American Mathematical Society*, 368(2):1321–1359, 2016. doi:10.1090/tran/6465.

19 Tobias Gärtner and Günter Hotz. Computability of analytic functions with analytic machines. In *Mathematical theory and computational practice*, volume 5635 of *Lecture Notes in Comput. Science*, pages 250–259. Springer, Berlin, 2009. doi:10.1007/978-3-642-03073-4_26.

20 Tobias Gärtner and Günter Hotz. Representation theorems for analytic machines and computability of analytic functions. *Theory of Computing Systems*, 51(1):65–84, 2012. doi:10.1007/s00224-011-9374-z.

21 Guido Gherardi and Alberto Marcone. How incomputable is the separable Hahn-Banach theorem? *Notre Dame Journal of Formal Logic*, 50(4):393–425, 2009. doi:10.1215/00294527-2009-018.

22 Eike Neumann and Arno Pauly. A topological view on algebraic computation models. arXiv 1602.08004, 2016. http://arxiv.org/abs/1602.08004.

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

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

25 Arno Pauly. The descriptive theory of represented spaces. arXiv 1408.5329, 2014. http://arxiv.org/abs/1408.5329.

26 Stephen G. Simpson. *Subsystems of Second Order Arithmetic*. Perspectives in Logic, Association for Symbolic Logic. Cambridge University Press, Poughkeepsie, second edition, 2009.

27 Thorsten von Stein. *Vergleich nicht konstruktiv lösbarer Probleme in der Analysis*. Fachbereich Informatik, FernUniversität Hagen, 1989. Diplomarbeit.

28 Klaus Weihrauch. *Computable Analysis*. Springer, Berlin, 2000.

29 Martin Ziegler. Real hypercomputation and continuity. *Theory of Computing Systems*, 41(1):177–206, 2007. doi:10.1007/s00224-006-1343-6.