Las Vegas Computability and Algorithmic Randomness*

Vasco Brattka[1,2], Guido Gherardi[2], and Rupert Hölzl[3]

1 **Department of Mathematics & Applied Mathematics,**
University of Cape Town,
South Africa,
Vasco.Brattka@cca-net.de

2 **Faculty of Computer Science,**
Universität der Bundeswehr München,
Germany,
Guido.Gherardi@gmail.com

3 **Department of Mathematics,**
National University of Singapore,
Republic of Singapore,
r@hoelzl.fr

Abstract

In this article we try to formalize the question "What can be computed with access to randomness?" We propose the very fine-grained Weihrauch lattice as an approach to differentiate between different types of computation with access to randomness. In particular, we show that a natural concept of Las Vegas computability on infinite objects is more powerful than mere oracle access to a Martin-Löf random object. As a concrete problem that is Las Vegas computable but not computable with access to a Martin-Löf random oracle we study the problem of finding Nash equilibria.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases Weihrauch degrees, weak weak Kőnig's lemma, Las Vegas computability, algorithmic randomness, Nash equilibria

Digital Object Identifier 10.4230/LIPIcs.STACS.2015.130

1 Introduction

Studying how access to sources of random information can enable or simplify the computation of certain mathematical objects is a recurring theme of theoretical computer science. This is particularly evident in the context of complexity theory, where for example the difficult question of whether P is equal to BPP has been studied for a long time—so far without an answer.

But the initial question can also be studied in more general settings. Many different versions have been studied in the field of algorithmic randomness. Here, the main subject is to find the correct formalization of what a random object is. The field has very strong interactions with the subject of computability theory (or, in the older terminology, recursion

* Vasco Brattka is supported by the National Research Foundation of South Africa. Rupert Hölzl was supported by a Feodor Lynen postdoctoral research fellowship of the Alexander von Humboldt Foundation and is supported by the Ministry of Education of Singapore through grant R146-000-184-112 (MOE2013-T2-1-062).

32nd Symposium on Theoretical Aspects of Computer Science (STACS 2015).
Editors: Ernst W. Mayr and Nicolas Ollinger; pp. 130–143

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

theory), and it has been extensively studied what computational properties random objects possess (see [17, 12] for recent monographs on algorithmic randomness).

It can be argued that this approach better represents the original question of what can be computed with access to randomness than, for example, the complexity-theoretic approach; the argument being that space or time bounds are not considered, meaning we are getting a better idea of the real computational *content* of random objects—as opposed to a gauge of their ability to *speed up* a computation until it can be performed within polynomial time. For this reason, the results from algorithmic randomness and computability theory certainly are of high importance. But in this article we will argue that one further ingredient is missing to capture best the intuitive idea behind the above question.

When thinking about the question in the setting of algorithmic randomness, maybe the first classic result that comes to mind is the following well-known theorem (this and the following Theorem 2 are discussed in [17, 12]).

▶ **Theorem 1** (Kučera-Gács Theorem). *Every sequence $A \in 2^{\mathbb{N}}$ is computable from some Martin-Löf random sequence $X \in 2^{\mathbb{N}}$.*

Informally this means that random objects compute everything. While it may seem at first that this result settles the initial question once and for all, it is not in fact a satisfactory answer. This is because it is not actually the randomness of X that is used to compute A; if this were the case then X could be replaced with any other "similarly random" set Y and it would still compute A. But this is not so: it is well known that the Turing upper cone of any non-computable set has measure 0 (we will discuss this in more detail in a moment).

This is a clear indication that in fact the computation of A does not really use the randomness of X. Instead, by studying the proof of the theorem, it becomes apparent that X is in fact "tailor-made" (or, perhaps, "tailor-chosen") to compute A. The randomness only enters the picture insofar as in order to be able to choose an X that computes A we need a large class of sets to choose from, and the class MLR happens to be large enough. Other sufficiently rich classes can be imagined where the same construction would be possible, a trivial example being the class of sets generated from Martin-Löf random sequences by inserting the symbol 0 in every second place.

So the Kučera-Gács Theorem does not settle our initial question, or only a very weak formalization of it. What we would rather like to know is what can be computed given access to *any* sufficiently random sequence. A second possible approach to the question might be Sacks' Measure Theorem.

▶ **Theorem 2** (Sacks' Measure Theorem). *If $A \in 2^{\mathbb{N}}$ is computable from every X in a subset of $2^{\mathbb{N}}$ of positive measure, then A is computable.*

Once again it might seem that this settles the question, as this states that randomness cannot compute anything of interest (i.e., that cannot be computed without randomness in any case). But again, this is not quite the answer to the question we are investigating. Sacks' theorem only applies if we want to compute a single set A. This is because the proof relies essentially on a majority vote argument.

But there are many very valid settings where this is not the case: often we are given a mathematical problem and want to find a solution to it, and we want to know whether randomness can help us to find such a solution. For a given instance of such a task there may be many admissible solutions; and each of these solutions may have a low probability of being produced by a Turing machine, so that a majority vote mechanic would fail.

To overcome this limitation we therefore need to work within a framework that is more involved while still holding on to the ideas of computability theory. This new framework is

provided by the Weihrauch degrees. The general idea here is to think about a mathematical task as a black box to which we pass an encoding of an instance of the mathematical problem at hand, and the black box has to return an encoding of *one of the* admissible solutions. One example might be a black box NASH which, given a bi-matrix game (A, B), will produce one of the possible Nash equilibria (x, y) of this game.

Once having established this framework we can now ask questions such as the following: Assume we have a black box solving a certain problem $\mathcal{A}$. Can we use it to solve another problem $\mathcal{B}$? The approach here is heavily inspired by many-one reducibility in computability theory, but applies (in general) to infinite objects instead of finite ones: we code an instance B of problem $\mathcal{B}$ into a valid instance A of problem $\mathcal{A}$. We then run the black box for $\mathcal{A}$ on A. The black box provides a solution for A. We then need to convert the solution for A back into a solution for B.

The reducibility sketched above is called *Weihrauch reducibility* and is *uniform*. That is, the pre-processing and post-processing steps in the above sketch are required to be performed by a pair of Turing machines uniformly for all instances of $\mathcal{B}$. The reducibility induces a rich lattice of mathematical problems of different difficulties.

In this article we will let $\mathcal{A}$ be a source of randomness, and then study which problems $\mathcal{B}$ are reducible to it. The randomness source $\mathcal{A}$ will be multi-valued, that is, while it is obliged to output random objects, it has free choice as to which specific random object it outputs. A problem $\mathcal{B}$ is only reducible to $\mathcal{A}$ if the pair of Turing machines is able to uniformly produce a solution for the given instances B of $\mathcal{B}$, no matter which choice the black box for $\mathcal{A}$ makes. The idea is that then the operations by the Turing machines can rely on no property of the black box's output *except on its randomness* to fulfill their task.[1] It is for this reason that we believe this approach correctly formalizes the initial question of "What can be computed with access to randomness?"

More concretely, we will identify two types of computations with access to random sources; we will determine their location in the Weihrauch lattice; and we will separate them by showing that one of them is significantly stronger than the other.

For the purposes of this extended abstract we focus on a number of key results and include only proof sketches; we also use a minimum of mathematical formalism. The present article focuses primarily on the interaction between computable analysis and algorithmic randomness; the results are presented in a form targeting a computer science audience. We rely to a large degree on computable analysis results from our forthcoming related article [7]. That article, in contrast to the present one, uses the full computable analysis formalism, presents the proofs of its results with all details, and contains much additional material.

2 The Weihrauch Lattice

The original definition of Weihrauch reducibility is due to Klaus Weihrauch [24] and has been studied for many years. More recently it has been noticed that a certain variant of this reducibility yields a lattice that is very suitable for the classification of the uniform

[1] For the sake of precision we should add that this idea is only perfectly realized for the randomness source MLR we study below; and only to a lesser degree by the randomness sources derived from WWKL, where the Turing machines actually can rely on some (very limited) additional property of the black box's output, namely the fact that it is guaranteed to be contained in some positive measure set. One could interpret the differences in computational strength of the different randomness sources studied in this article as stemming from this distinction.

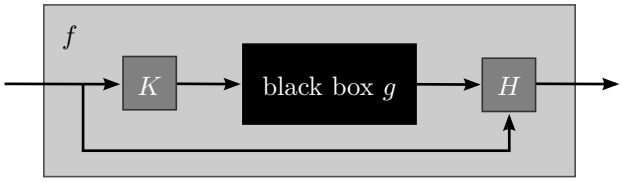

Figure 1 A visualization of the Weihrauch reducibility of f to g via Turing machines K and H.

computational content of mathematical problems (see [13, 20, 19, 6, 5, 4, 8, 11]). The basic reference for all notions from computable analysis is Weihrauch's textbook [25].

Formally, the Weihrauch lattice is formed by equivalence classes of partial multi-valued functions $f :\subseteq X \rightrightarrows Y$ on represented spaces X, Y. A multi-valued function is considered as a computational problem in the sense that for every $x \in \mathrm{dom}(f)$ the goal is to find *some* $y \in f(x)$. A typical mathematical problem f would be the problem of solving some type of equation or to determine Nash equilibria, as mentioned above. In this case $\mathrm{dom}(f)$ would contain the admissible instances x of the problem and for each instance x the set $f(x)$ would contain the corresponding solutions.

A *represented space* (X, δ) is a set X together with a surjective partial map $\delta :\subseteq \mathbb{N}^{\mathbb{N}} \to X$ 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. In order to keep the presentation here as accessible as possible, we will not make any technical use of the machinery of representations and we refer the reader to [7] for all details.

The intuition behind Weihrauch reducibility is that $f \leq_{\mathrm{W}} g$ holds if there is a computational procedure for solving f when allowed a single application of the computational resource g. We make this somewhat more precise.

▶ **Definition 3** (Weihrauch reducibility)**.** Let $f :\subseteq X \rightrightarrows Y$ and $g :\subseteq W \rightrightarrows Z$ be multi-valued functions on represented spaces. Then f is called *Weihrauch reducible* to g, in symbols $f \leq_{\mathrm{W}} g$, if f can be simulated through an oracle computational process that accesses the oracle g exactly once; that is, if given a name $\widehat{x}$ of an input $x \in \mathrm{dom}(f)$ it is effectively possible to compute a name $\widehat{y}$ of some value $y \in f(x)$ consulting the oracle function g exactly on one input $w \in \mathrm{dom}(g)$. Formally, the computation works by having a pair of two Turing machines K and H. K operates on $\widehat{x}$ to generate $\widehat{w}$, a name for w. The result z of the evaluation of g on w will again be represented by a name $\widehat{z}$. Then the second Turing machine H operates on $\widehat{z}$ to generate $\widehat{y}$, the name representing the final output y.

The concept of Weihrauch reducibility can be seen as a variant of many-one reducibility for multi-valued functions on infinite objects. The informal definition above is equivalent to the more formal definition in [7] via [23, Theorem 7.2].

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

- $f \times g$ 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.
- $f \sqcup g$ is the *coproduct* of f and g and represents the alternative evaluation of f on some input x or g on some input w (where the input set is the disjoint union $X \sqcup W$ and the output set is $Y \sqcup Z$).

- If $Z = X$, then $f \circ g :\subseteq W \rightrightarrows Y$ is the composition of f and g. Note that since f and g can be partial the following careful definition is needed.

$$\begin{array}{rcl} \mathrm{dom}(f \circ g) & := & \{w \in W : g(w) \subseteq \mathrm{dom}(f)\} \text{ and} \\ (f \circ g)(w) & := & \{y \in Y : (\exists x \in X)(y \in f(x) \text{ and } x \in g(w))\}. \end{array}$$

- $f * g := \sup\{f_0 \circ g_0 : f_0 \leq_{\mathrm{W}} f \text{ and } g_0 \leq_{\mathrm{W}} g\}$ 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 $n \in \mathbb{N}$.

The coproduct $f \sqcup g$ is the supremum in the Weihrauch lattice (and we do not specify the infimum operation here). The lattice is not complete as infinite suprema do not need to exist, but the supremum $f * g$ always exists. The finite parallelization is a closure operator in the Weihrauch lattice. Further details can be found in [7].

3 Las Vegas Computability

Randomized algorithms have been studied in the discrete setting for a long time (see for instance Motwani and Raghavan [15]), but very little is known for computations on infinite objects. We will now present a formalization of Las Vegas computability with probabilistic Turing machines that is very close to Babai's original understanding of this concept [3].

▶ **Definition 4** (Las Vegas computability)**.** A partial multi-valued function $f :\subseteq X \rightrightarrows Y$ on represented spaces is called *Las Vegas computable*, if there exists a Turing machine M that given some input $p \in \mathbb{N}^{\mathbb{N}}$ and some auxiliary input $r \in 2^{\mathbb{N}}$ (the "random advice") computes f in the following way. The machine M either produces some infinite output $q \in \mathbb{N}^{\mathbb{N}}$ or stops after a finite number of steps and signals a failure with the following additional constraints:

1. if p is a name of some input $x \in \mathrm{dom}(f)$ and M does not fail, then q has to be a name of a correct output $y \in f(x)$,
2. for any fixed such p there is a positive probability that for some random advice r the machine M does not fail.

The aforementioned condition can be made precise by requiring that the set S_p of successful random advices r for which the machine computes without failure has positive measure $\mu(S_p) > 0$ for every fixed p that is a name of some $x \in \mathrm{dom}(f)$. Here μ denotes the uniform measure on $2^{\mathbb{N}}$.

The essential feature of a Las Vegas computation is that it produces a result with positive probability, and if it produces a result, then this result is correct. Hence, the correctness of the computation is never compromised, only the success of the computation is subject to randomization.

We mention that Las Vegas computability as defined above is a refinement of the definition of non-deterministically computable functions, as originally introduced by Martin Ziegler [26] and further studied in [4]. The difference between Las Vegas computability and non-deterministic computability is that in the former case one asks for a positive probability that a piece of advice is successful whereas in the latter case one just asks for the existence of a successful piece of advice.

We now show that the class of Las Vegas computable functions is closed under composition, which means that this class is "reasonable" in a certain sense. The proof of the following theorem is a refined version of the corresponding proof for non-deterministic functions in [4] with an additional invocation of Fubini's Theorem.

▶ **Theorem 5** (Independent Choice). *The class of Las Vegas computable multi-valued functions on represented spaces is closed under composition.*

Proof sketch. Let $f :\subseteq Y \rightrightarrows Z$ and $g :\subseteq X \rightrightarrows Y$ be Las Vegas computable, witnessed by two probabilistic Turing machines M_f and M_g. We describe the construction of a suitable probabilistic Turing machine M for $f \circ g$. Given a name p of some $x \in \mathrm{dom}(f \circ g)$ as an input for M, we just interpret the random advice of M as a pair $\langle r, s\rangle \in 2^{\mathbb{N}}$ and we use s as random advice in order to simulate M_g on input p and then we simulate M_f on the corresponding output with advice r. The composed machine M fails if either of the simulations of the two machines M_g or M_f fails in this process. It is clear that the composed machine will produce a correct result for $f \circ g$ if it does not fail. We now have to look at the success probabilities of the corresponding machines, i.e., at the measures of the following sets:

- S_p is the set of successful advices s of M_g on input p;
- $R_{p,s}$ is the set of successful advices r of M_f on the output q that M_g produces on input p with advice s;
- T_p is the set of successful advices $\langle r, s\rangle$ of M on input p.

We know that $\mu(S_p) > 0$ for all p that are names of elements of $\mathrm{dom}(g)$ and $\mu(R_{p,s}) > 0$ for all combinations of p, s that lead to an output q of M_g that is the name of an element of $\mathrm{dom}(f)$. We need to prove $\mu(T_p) > 0$ for all p that are names of elements of $\mathrm{dom}(f \circ g)$. We obtain

$$T_p = \{\langle r, s\rangle \in 2^{\mathbb{N}} : s \in S_p \text{ and } r \in R_{p,s}\},$$

which implies by the Theorem of Fubini for measurable sets and (strict) monotonicity of the integral for non-negative functions that

$$\mu(T_p) = \int_{S_p} \mu(R_{p,s})\,\mathrm{d}\mu > 0.$$

Here the integrand is understood to be the function $s \mapsto \mu(R_{p,s})$. This proves that M satisfies the necessary conditions. ◀

4 Weak Weak Kőnig's Lemma

In this section we will see that we can also characterize Las Vegas computability with the help of Weihrauch reducibility and Weak Weak Kőnig's Lemma. Weak Kőnig's Lemma is a principle that has been intensively studied in reverse mathematics [22]. 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 $\mathsf{WKL} :\subseteq \mathrm{Tr} \rightrightarrows 2^{\mathbb{N}}, T \mapsto [T]$ that maps an infinite binary tree $T \subseteq 2^{<\mathbb{N}}$ 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. In [4] it was proved that

$$f \leq_{\mathrm{W}} \mathsf{WKL} \iff f \text{ is non-deterministically computable,}$$

and here we prove a similar result for the so-called Weak Weak Kőnig's Lemma and Las Vegas computability. Weak Weak Kőnig's Lemma has also been introduced in reverse mathematics and for us it is the problem to find an infinite path in a binary tree of positive measure, i.e., $\mathsf{WWKL} :\subseteq \mathrm{Tr} \rightrightarrows 2^{\mathbb{N}}, T \mapsto [T]$, which is the restriction of WKL to the set $\mathrm{dom}(\mathsf{WWKL}) := \{T \in \mathrm{Tr} : \mu([T]) > 0\}$. Using this notation we obtain the following result.

▶ **Theorem 6** (Las Vegas computability). *For $f :\subseteq X \rightrightarrows Y$ we obtain:*

$$f \leq_{\mathrm{W}} \mathsf{WWKL} \iff f \text{ is Las Vegas computable.}$$

Proof sketch. There is a computable map from infinite binary trees to closed sets given by $T \mapsto [T]$ and this map has a computable multi-valued right inverse. Here trees $T \subseteq 2^{<\mathbb{N}}$ are represented via their characteristic functions $\mathrm{cf}_T : 2^{<\mathbb{N}} \to \{0,1\}$ and closed sets $A \subseteq 2^{\mathbb{N}}$ are represented by negative information, for instance by an enumeration of sufficiently many balls $w2^{\mathbb{N}}$ that exhaust the complement of A. Now the fact that f is Las Vegas computable means that f can be computed by a probabilistic machine that takes advantage of a random input $r \in 2^{\mathbb{N}}$. For any fixed input p this machine can also identify the unsuccessful advices, which means that the set $S_p \subseteq 2^{\mathbb{N}}$ of successful advices is co-c.e. closed in p, i.e., we can compute it with respect to negative information. This information can be converted into the characteristic function of a tree T with $[T] = S_p$ and this yields a reduction $f \leq_{\mathrm{W}} \mathsf{WWKL}$, since $\mu(S_p) > 0$. Vice versa any such reduction can be converted into a corresponding Las Vegas machine. ◀

Using a result of Jockusch and Soare that generalizes Theorem 2, one can prove $\mathsf{WKL} \not\leq_{\mathrm{W}} \mathsf{WWKL}$ (see [10, Theorem 20]). This means that WWKL is strictly below WKL.

▶ **Proposition 7.** $\mathsf{WWKL} <_{\mathrm{W}} \mathsf{WKL}$.

We can also rephrase this result as follows.

▶ **Corollary 8.** *Every Las Vegas computable multi-valued function is non-deterministically computable, but there are non-deterministically computable multi-valued functions that are not Las Vegas computable.*

In [7] we prove (with a finite extension argument) that determining zeros of continuous functions with sign changes is a concrete problem that is non-deterministically computable but not Las Vegas computable. We close this section by mentioning another important observation.

▶ **Proposition 9.** $\mathsf{WWKL} \equiv_{\mathrm{W}} \mathsf{WWKL}^*$.

Proof. The reduction $\mathsf{WWKL} \leq_{\mathrm{W}} \mathsf{WWKL}^*$ is obvious. For the other direction, notice that for any two multi-valued functions f and g we have $f \times g = (\mathrm{id} \times g) \circ (f \times \mathrm{id})$ where $f \times \mathrm{id} \leq_{\mathrm{W}} f$ and $\mathrm{id} \times g \leq_{\mathrm{W}} g$. Then Theorems 6 and 5 applied to $f = g = \mathsf{WWKL}$ show $\mathsf{WWKL}^2 \leq_{\mathrm{W}} \mathsf{WWKL}$. Iteration of this argument concludes the proof. ◀

5 Dependence on the Probability

Next one could ask whether there is a difference between the setting described so far (i.e., the setting where we only demand positivity of the success probabilities $\mu(S_p)$) and a setting where one demands fixed minimum probabilities $\mu(S_p) > \varepsilon$ for some $\varepsilon > 0$.

This question can and has already been studied in the form of a corresponding restriction of Weak Weak Kőnig's Lemma: Dorais et al. [11] have introduced the problem ε-WWKL, which is WWKL restricted to $\mathrm{dom}(\varepsilon\text{-}\mathsf{WWKL}) := \{T \in \mathrm{Tr} : \mu([T]) > \varepsilon\}$, i.e., to trees whose sets of infinite paths have measure larger than ε. It is clear that if the lower probability bound ε decreases, then one can compute at least as much as before, i.e., the map $\varepsilon \mapsto \varepsilon\text{-}\mathsf{WWKL}$ is anti-monotone with regards to Weihrauch reducibility. We have proved that it is even strictly anti-monotone, a result that has independently been obtained by Dorais et al. [11]. Our proof

is essentially a combination of a combinatorial argument that is based on a certain version of the pigeonhole principle, combined with a topological and measure-theoretic reasoning. We just sketch the idea.

▶ **Theorem 10.** $\varepsilon \geq \delta \iff \varepsilon\text{-}\mathsf{WWKL} \leq_{\mathrm{W}} \delta\text{-}\mathsf{WWKL}$ *for* $\varepsilon, \delta \in [0,1]$.

Proof idea. We only need to prove "$\Longleftarrow$". Let $\varepsilon < \delta$. Then there are positive integers $a < b$ with $\varepsilon < \frac{a}{b} < \delta$. We consider the problem $\mathsf{C}_{a,b}$ of finding a point in a closed subset $A \subseteq \{0, ..., b-1\}$ (given by negative information) of cardinality $|A| \geq a$. One can easily prove that $\mathsf{C}_{a,b} \leq_{\mathrm{W}} \varepsilon\text{-}\mathsf{WWKL}$; and a more involved argument based on a corresponding pigeonhole principle shows $\mathsf{C}_{a,b} \not\leq_{\mathrm{W}} \delta\text{-}\mathsf{WWKL}$. This proves $\varepsilon\text{-}\mathsf{WWKL} \not\leq_{\mathrm{W}} \delta\text{-}\mathsf{WWKL}$. ◀

The aforementioned result can be interpreted such that probability amplification fails for Las Vegas computable functions. Intuitively, this is because we are dealing with infinite computations and even if we perform two randomized computations in parallel we need to start producing some definite output without ever knowing whether one of the involved computations might turn out to be a failure at some later stage.

At one extreme end of the probability spectrum we have $0\text{-}\mathsf{WWKL}$, which is identical to WWKL and hence it represents Las Vegas computations. On the other hand, we have $1\text{-}\mathsf{WWKL}$, which is easily seen to be computable (since every *closed* set $A \subseteq 2^{\mathbb{N}}$ of measure 1 needs to be the full space $A = 2^{\mathbb{N}}$). However, there is still a non-computable problem below all the $\varepsilon\text{-}\mathsf{WWKL}$ with $\varepsilon \in [0,1)$ that is of interest to us, and in a certain sense it is the infimum of all problems $\varepsilon\text{-}\mathsf{WWKL}$: this is $(1-*)\text{-}\mathsf{WWKL} :\subseteq \mathrm{Tr}^{\mathbb{N}} \rightrightarrows 2^{\mathbb{N}}$, defined by

$$(1-*)\text{-}\mathsf{WWKL}((T_n)_{n\in\mathbb{N}}) := \bigsqcup_{n\in\mathbb{N}} (1-2^{-n})\text{-}\mathsf{WWKL}(T_n),$$

where $\mathrm{dom}((1-*)\text{-}\mathsf{WWKL}) := \{(T_n)_{n\in\mathbb{N}} \in \mathrm{Tr}^{\mathbb{N}} : (\forall n \in \mathbb{N})\ \mu([T_n]) > 1-2^{-n}\}$. Intuitively, this problem is the following: given a sequence of infinite binary trees $(T_n)_{n\in\mathbb{N}}$ with $\mu([T_n]) > 1-2^{-n}$ for all n, we want to find one infinite path $p \in [T_n]$ in *one* of the trees T_n. One easily obtains the following corollary from Theorem 10.

▶ **Corollary 11.** $(1-*)\text{-}\mathsf{WWKL} <_{\mathrm{W}} \varepsilon\text{-}\mathsf{WWKL}$ *for every* $\varepsilon \in [0,1)$.

6 Algorithmic Randomness

With the arguments in the previous sections we were able to identify the Weihrauch degree of WWKL as that of a natural kind of randomized computation. Having done this, we are now able to locate this type of randomized computation in the Weihrauch lattice and to compare it with other types. Another natural type is computation with access to Martin-Löf oracles. We recall that this has been extensively studied from a non-uniform perspective in computability theory (see [17, 12]). But of course, here we will again take the Weihrauch lattice perspective: we ask what can be reduced to the principle $\mathsf{MLR} :\subseteq 2^{\mathbb{N}} \rightrightarrows 2^{\mathbb{N}}$, which maps an arbitrary input $x \in 2^{\mathbb{N}}$ to an output $y \in 2^{\mathbb{N}}$ that is Martin-Löf random relative to x. In fact, we call the functions $f :\subseteq X \rightrightarrows Y$ with

$$f \leq_{\mathrm{W}} \mathsf{MLR}$$

Martin-Löf computable and they can be seen as those functions that are computable on a *Martin-Löf machine*, i.e., on a machine that can request a Martin-Löf random sequence (relative to the input) exactly once during the course of its computation. Now the obvious question is: how does the power of Martin-Löf machines compare to Las Vegas machines?

We will see that in the Weihrauch lattice MLR is strictly weaker than WWKL and that the distance in the lattice is in fact quite large. This follows from the following result.

▶ **Theorem 12** (Martin-Löf computability)**.** $\mathsf{MLR} <_{\mathrm{W}} (1-*)\text{-}\mathsf{WWKL}$.

Proof sketch of Theorem 12. We recall (see [17, 12]) that there is a universal Martin-Löf test, which is a computable sequence $(U_i)_{i\in\mathbb{N}}$ of c.e. open sets $U_i \subseteq 2^{\mathbb{N}}$ such that $\mu(U_i) < 2^{-n}$ and $\bigcap_{i=0}^{\infty} U_i$ is exactly the set of all sequences which are not Martin-Löf random. Hence, each complement $A_i := 2^{\mathbb{N}} \setminus U_i$ is a co-c.e. closed set with $\mu(A_i) > 1 - 2^{-n}$ and each A_i only contains Martin-Löf random sequences. Hence, we can compute a corresponding sequence $(T_i)_{i\in\mathbb{N}}$ of infinite binary trees with $[T_i] = A_i$. Upon input of this sequence, $(1-*)$-WWKL yields a Martin-Löf random sequence. The entire argument can be relativized, i.e., it also works in the presence of some oracle $p \in 2^{\mathbb{N}}$. This yields the reduction $\mathsf{MLR} \leq_{\mathrm{W}} (1-*)\text{-}\mathsf{WWKL}$, and this reduction is strict according to [9, Lemma 7.4]. ◀

Together with Corollary 11 the previous theorem yields the following statement which constitutes one of the central results of this article.

▶ **Corollary 13.** *Every Martin-Löf computable multi-valued function is also Las Vegas computable, but there are Las Vegas computable multi-valued functions which are not Martin-Löf computable.*

Notice the stark contrast with reverse mathematics, where the principles WWKL and MLR are equivalent over RCA_0. This follows essentially from a formalization of a theorem of Kučera [14]; a detailed alternate proof, which even works over the weaker proof system RCA_0^*, has been given by Avigad et al. [2, Theorem 3.1].

In the next section we will see a concrete example of a problem that is Las Vegas computable, but not Martin-Löf computable.

7 Nash Equilibria

To show that Las Vegas computability is more than just a purely theoretical notion, we will give a concrete example of a useful mathematical task that can be performed with it but not with the weaker types of randomized computation studied in this article.

To this end we have proved (based on results of Arno Pauly) that there is a Las Vegas algorithm to compute Nash equilibria. We recall from [19, 18] that a pair $A, B \in \mathbb{R}^{m\times n}$ of $m \times n$–matrices is called a *bi-matrix game*. Any vector $s = (s_1, ..., s_m) \in \mathbb{R}^m$ with $s_i \geq 0$ for all $i = 1, ..., m$ and $\sum_{j=1}^{m} s_j = 1$ is called a *mixed strategy*. By S^m we denote the set of these mixed strategies of dimension m. Then a *Nash equilibrium* is a pair $(x, y) \in S^n \times S^m$ of strategies such that

1. $x^{\mathrm{T}} A y \geq w^{\mathrm{T}} A y$ for all $w \in S^n$ and
2. $x^{\mathrm{T}} B y \geq x^{\mathrm{T}} B z$ for all $z \in S^m$.

Nash [16] proved that for any bi-matrix game there exists a Nash equilibrium. By $\mathsf{NASH}_{n,m} : \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n} \rightrightarrows \mathbb{R}^n \times \mathbb{R}^m$ we denote the corresponding problem

$$\mathsf{NASH}_{n,m}(A, B) := \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^m : (x, y) \text{ is a Nash equilibrium for } (A, B)\}$$

of finding a Nash-equilibrium for an $m \times n$ bi-matrix game and by $\mathsf{NASH} := \bigsqcup_{n,m\in\mathbb{N}} \mathsf{NASH}_{n,m}$ we denote the coproduct of all such games for finite $m, n \in \mathbb{N}$. This means intuitively that NASH is the following problem: given a bi-matrix game (A, B) of arbitrary known dimension

$m \times n$, find a Nash equilibrium (x, y) of (A, B). Pauly [19, 21] proved the following theorem asserting that the problem NASH is Weihrauch equivalent to the idempotent closure of robust division RDIV on the unit interval.

▶ **Theorem 14** (Nash equilibria). $\mathsf{NASH} \equiv_{\mathrm{W}} \mathsf{RDIV}^*$.

Robust division is the multi-valued function $\mathsf{RDIV} : [0,1]^2 \rightrightarrows [0,1]$ with

$$\mathsf{RDIV}(x,y) := \begin{cases} \{\frac{x}{\max(x,y)}\} & \text{if } y > 0, \\ [0,1] & \text{otherwise.} \end{cases}$$

In other words, RDIV is essentially the problem of computing the fraction $\frac{x}{y}$ within $[0,1]$, where the result is allowed to be arbitrary in case that the denominator y is zero. It is easy to see that RDIV is discontinuous and hence not computable. The result $\mathsf{NASH} \equiv_{\mathrm{W}} \mathsf{RDIV}^*$ means that one can compute Nash equilibria with a certain number of parallel robust divisions (where the number of divisions needed is known a priori). The intuition behind this is that robust division can be used to solve linear equations and linear inequalities in a compact domain and repeated operations of this type can lead (in a rather involved way) to a Nash equilibrium. In order to prove that Nash equilibria are Las Vegas computable it suffices by Proposition 9 to show that RDIV is Las Vegas computable, i.e., $\mathsf{RDIV} \leq_{\mathrm{W}} \mathsf{WWKL}$.

▶ **Proposition 15.** $\mathsf{RDIV} \leq_{\mathrm{W}} \mathsf{WWKL}$.

We do not give a full proof but describe the idea behind the Las Vegas algorithm for robust division RDIV informally (we note that $r \in [0,1]$ can be used equivalently instead of $r \in 2^{\mathbb{N}}$ as random advice):

1. Given $x, y \in [0,1]$ and a random advice $r \in [0,1]$, we aim to compute the fraction $z = \frac{x}{\max(x,y)}$.
2. We guess that r is a correct solution, in particular $r = z$ if $y > 0$, and we start to output longer and longer approximations of r (in form of rational intervals $[a,b]$ with $a < r < b$).
3. Simultaneously, we try to find out whether $y > 0$, which we will eventually recognize, if this is correct.
4. As soon as we find that $y > 0$, we can compute the true result $z = \frac{x}{\max(x,y)}$ and in this case we start to produce approximations of z as output.
5. If at some stage we find that the best approximation $[a,b]$ of r that was already produced as output is incompatible with z, i.e., if $z \notin [a,b]$, then we stop the computation and indicate that it failed.

This algorithm produces a correct result $z \in \mathsf{RDIV}(x,y)$ whenever it does not fail. It can only fail if $y > 0$. In the moment when the algorithm detects $y > 0$, then there is still a positive probability $\mu([a,b]) = b - a > 0$ that the random advice r, which has only been approximated up to $[a,b]$ so far, is compatible with the true result $z = \frac{x}{\max(x,y)}$. Hence, the above algorithm constitutes a Las Vegas algorithm for robust division. Altogether we obtain the following.

▶ **Corollary 16.** $\mathsf{NASH} \leq_{\mathrm{W}} \mathsf{WWKL}$.

The above algorithm for robust division only yields success probabilities arbitrarily close to zero (depending on when it is recognized that the denominator is positive). In fact, one can prove that robust division (and hence Nash equilibria) cannot be computed with any fixed positive success probability.

▶ **Proposition 17.** $\mathsf{RDIV} \not\leq_{\mathrm{W}} \varepsilon\text{-}\mathsf{WWKL}$ *for* $\varepsilon > 0$.

In particular, this implies the following by Corollary 11 and Theorem 12.

▶ **Corollary 18.** *Computing Nash equilibria is Las Vegas computable but not Martin-Löf computable.*

Also note that Proposition 17 implies that the reducibility in Corollary 16 is strict.

8 Conclusions

In this paper we have introduced the class of Las Vegas computable functions (for computations with infinite objects), and we have proved that it is strictly included in between the classes of Martin-Löf computable functions and non-deterministically computable functions. As a natural example of a Las Vegas computable problem that is not Martin-Löf computable, we have discussed the problem of finding Nash equilibria. Principles very closely related to MLR and WWKL turned out to be equivalent to each other in the non-uniform and more coarse-grained setting of reverse mathematics [14, 2]. Hence, reverse mathematics could not uncover the fine-grained uniform distinctions that we have made with the help of the Weihrauch lattice.

Having presented our above results, we should at the closure of this article mention other existing work in which the question "What can be computed with access to randomness?" was studied in other contexts, and clarify how these settings differ from ours. The work concerning the complexity-theoretic setting may be the most well-known one. There, similar motivations to the ones behind our present article have led, among other questions, to the study of the inclusion chain $\mathsf{P} \subseteq \mathsf{ZPP} \subseteq \mathsf{BPP}$. The central open question in this area is whether $\mathsf{P} = \mathsf{BPP}$. The setting differs very much from ours: First of all, it is discrete, that is, the objects computed are finite strings. Furthermore, resource bounds are present in this setting; they are in fact required to make this study interesting, as without them there is no computability-theoretic difference between deterministic and randomized computation.

Note that there is also a body of work by Allender et al. in the complexity-theoretic setting (see, for example, [1]). While it may seem closely related to the initial question at first, we would like to point out that the computations studied there are not truly taking advantage of randomness as a resource for computation. Rather, in this work, computations are made using reductions to the sets R_{C}, R_{K}, and so on. These are the sets containing the information about which finite strings are compressible and which are not; therefore the work by Allender et al. can be thought of as not being about computation from *randomness*, but rather about computation from the *knowledge about what is* random.

Once we leave the domain of complexity theory, that is, give up resource bounds on the computations, we need to look at the computation of infinite objects if we want to find interesting insights into the power of randomness. In this area the important distinction is then that between uniform and non-uniform computation. This paper has focused on the levels of computation power that are needed in the uniform, infinite setting. If one wants to look at the *non*-uniform counterpart of our work then that is the work on WWKL and MLR in reverse mathematics, as cited above.

Acknowledgments. The authors would like to thank the referees for their detailed feedback.

References

1 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM Journal on Computing*, 35(6):1467–1493, 2006.

2 Jeremy Avigad, Edward T. Dean, and Jason Rute. Algorithmic randomness, reverse mathematics, and the dominated convergence theorem. *Annals of Pure and Applied Logic*, 163(12):1854–1864, 2012.

3 László Babai. Monte-Carlo algorithms in graph isomorphism testing. Technical Report No. 79-10, Université de Montréal, Département de Mathématique et de Statistique, 1979.

4 Vasco Brattka, Matthew de Brecht, and Arno Pauly. Closed choice and a uniform low basis theorem. *Annals of Pure and Applied Logic*, 163(8):986–1008, 2012.

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

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

7 Vasco Brattka, Guido Gherardi, and Rupert Hölzl. Probabilistic computability and choice. July 2014. Preliminary version available at http://arxiv.org/abs/1312.7305.

8 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(6):623–655, 2012.

9 Vasco Brattka, Matthew Hendtlass, and Alexander P. Kreuzer. On the uniform computational content of computability theory. January 2015. Preliminary version available at http://arxiv.org/abs/1501.00433.

10 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.

11 François G. Dorais, Damir D. Dzhafarov, Jeffry L. Hirst, Joseph R. Mileti, and Paul Shafer. On uniform relationships between combinatorial problems. *Transactions of the AMS*, 2014. Accepted for publication. Preliminary version available at http://arxiv.org/abs/1212.0157.

12 Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic randomness and complexity*. Theory and Applications of Computability. Springer, New York, 2010.

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

14 Antonín Kučera. Measure, Π^0_1-classes and complete extensions of PA. In Heinz-Dieter Ebbinghaus, Gert H. Müller, and Gerald E. Sacks, editors, *Recursion Theory Week. Proceedings of the Conference Held at the Mathematisches Forschungsinstitut in Oberwolfach, April 15–21, 1984*, volume 1141 of *Lecture Notes in Mathematics*, pages 245–259. Springer, Berlin, 1985.

15 Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge, 1995.

16 John Nash. Non-cooperative games. *Annals of Mathematics*, 54:286–295, 1951.

17 André Nies. *Computability and Randomness*, volume 51 of *Oxford Logic Guides*. Oxford University Press, New York, 2009.

18 Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, Cambridge, 2007.

19 Arno Pauly. How incomputable is finding Nash equilibria? *Journal of Universal Computer Science*, 16(18):2686–2710, 2010.

20 Arno Pauly. On the (semi)lattices induced by continuous reducibilities. *Mathematical Logic Quarterly*, 56(5):488–502, 2010.

21 Arno Pauly. *Computable Metamathematics and its Application to Game Theory*. PhD thesis, University of Cambridge, Computer Laboratory, Clare College, Cambridge, 2011.

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

23 Nazanin R. Tavana and Klaus Weihrauch. Turing machines on represented sets, a model of computation for analysis. *Logical Methods in Computer Science*, 7(2:19):1–21, 2011.

24 Klaus Weihrauch. The degrees of discontinuity of some translators between representations of the real numbers. Technical Report TR-92-050, International Computer Science Institute, Berkeley, July 1992.

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

26 Martin Ziegler. Real hypercomputation and continuity. *Theory of Computing Systems*, 41(1):177–206, 2007.

Revision Notice

This is a revised version of the eponymous paper appeared in the proceedings of STACS 2015 (LIPIcs, volume 30, http://www.dagstuhl.de/dagpub/978-3-939897-78-1, published in March, 2015), in which an incorrect class inclusion was removed from the inclusion chain in Section 8.

Dagstuhl Publishing – April 15, 2015.