Conference contribution :
Kihara, T. & Pauly, A. (2019). *Finite choice, convex choice and sorting.* Theory and Applications of Models of Computation, (pp. 378-393). Japan: 15th Annual Conference, TAMC 2019, Kitakyushu, Japan, April 13–16, 2019.
http://dx.doi.org/10.1007/978-3-030-14812-6

Finite choice, convex choice and sorting*

Takayuki Kihara[1] and Arno Pauly[2,3][0000−0002−0173−3295]

[1] Department of Mathematical Informatics
Nagoya University, Nagoya, Japan
kihara@i.nagoya-u.ac.jp
[2] Department of Computer Science
Swansea University, Swansea, UK
[3] Department of Computer Science
University of Birmingham, Birmingham, UK
Arno.M.Pauly@gmail.com

Abstract. We study the Weihrauch degrees of closed choice for finite sets, closed choice for convex sets and sorting infinite sequences over finite alphabets. Our main result is that choice for finite sets of cardinality $i+1$ is reducible to choice for convex sets in dimension j, which in turn is reducible to sorting infinite sequences over an alphabet of size $k+1$, iff $i \leq j \leq k$. Our proofs invoke Kleene's recursion theorem, and we describe in some detail how Kleene's recursion theorem gives rise to a technique for proving separations of Weihrauch degrees.

Keywords: computable analysis · Weihrauch reducibility · Closed choice.

1 Introduction

The Weihrauch degrees are the degrees of non-computability for problems in computable analysis. In the wake of work by Brattka, Gherardi, Marcone and P. [16, 4, 3, 24] they have become a very active research area in the past decade. A recent survey is found as [7].

We study the Weihrauch degrees of closed choice for finite sets, closed choice for convex sets and sorting infinite sequences over finite alphabets. The closed choice operators have turned out to be a useful scaffolding in that structure: We often classify interesting operations (for example linked to existence theorems) as being equivalent to a choice operator, and then prove separations for the choice operators, as they are particularly amenable for many proof techniques. Examples of this are found in [3, 2, 11, 5, 9, 6, 10, 21, 17]. Convex choice in particular

* Kihara's research was partially supported by JSPS KAKENHI Grant 17H06738, 15H03634, and the JSPS Core-to-Core Program (A. Advanced Research Networks).

* This project has received funding from the European Unions Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 731143, *Computing with Infinite Data*.

captures the degree of non-computability of finding fixed points of non-expansive mappings via the Goehde-Browder-Kirk fixed point theorem [21].

The present article is a continuation of [20] by Le Roux and P., which already obtained some results on the connections between closed choice for convex sets and closed choice for finite sets. We introduce new proof techniques and explore the connection to the degree of sorting infinite sequences. Besides laying the foundations for future investigations of specific theorems, we are also addressing a question on the complexity caused by dimension: Researchers have often wondered whether there is a connection between the dimension of the ambient space and the complexity of certain choice principles. An initial candidate was to explore closed choice for connected subsets, but it turned out that the degree is independent of the dimension, provided this is at least 2 [10]. As already shown in [20], this works for convex choice. One reason for this was already revealed in [20]: We need n dimensions in order to encode a set of cardinality $n+1$. We add another reason here: Each dimension requires a separate instance of sorting an infinite binary sequence in order to find a point in a convex set.

Structure of the paper Most of our results are summarized in Figure 1 on Page 5. Section 2 provides a brief introduction to Weihrauch reducibility. In Section 3 we provide formal definitions of the principles under investigation, and give a bit more context. We proceed to introduce our new technique to prove separations between Weihrauch degrees in Section 4; it is based on Kleene's recursion theorem. The degree of sorting an infinite binary sequence is studied in Section 5, including a separation technique adapted specifically for this in Subsection 5.1, its connection to convex choice in Subsection 5.2 and a digression on the task of finding connected components of countable graphs in Subsection 5.3. Section 6 is constituted by Theorem 5 and its proof, establishing the precise relationship between finite choice and sorting. Finally, in Section 7 we introduce a game characterizing reducibility between finite choice for varying cardinalities.

2 Background on Weihrauch reducibility

Weihrauch reducibility is a quasiorder defined on multi-valued functions between represented spaces. We only give the core definitions here, and refer to [25] for a more in-depth treatment. Other sources for computable analysis are [29, 8].

Definition 1. *A represented space* $\mathbf{X}$ *is a set* X *together with a partial surjection* $\delta_\mathbf{X} :\subseteq \mathbb{N}^\mathbb{N} \to X$.

A partial function $F :\subseteq \mathbb{N}^\mathbb{N} \to \mathbb{N}^\mathbb{N}$ is called a *realizer* of a function $f :\subseteq \mathbf{X} \to \mathbf{Y}$ between represented spaces, if $f(\delta_\mathbf{X}(p)) = \delta_\mathbf{Y}(F(p))$ holds for all $p \in \mathrm{dom}(f \circ \delta_\mathbf{X})$. We denote F being an realizer of f by $F \vdash f$. We then call $f :\subseteq \mathbf{X} \to \mathbf{Y}$ *computable* (respectively *continuous*), iff it has a computable (respectively continuous) realizer.

Represented spaces can adequately model most spaces of interest in *everyday mathematics*. For our purposes, we are primarily interested in the construction of the hyperspace of closed subsets of a given space.

The category of represented spaces and continuous functions is cartesian-closed, by virtue of the UTM-theorem. Thus, for any two represented spaces $\mathbf{X}$, $\mathbf{Y}$ we have a represented spaces $\mathcal{C}(\mathbf{X}, \mathbf{Y})$ of continuous functions from $\mathbf{X}$ to $\mathbf{Y}$. The expected operations involving $\mathcal{C}(\mathbf{X}, \mathbf{Y})$ (evaluation, composition, (un)currying) are all computable.

Using the Sierpiński space $\mathbb{S}$ with underlying set $\{\top, \bot\}$ and representation $\delta_\mathbf{S} : \mathbb{N}^\mathbb{N} \to \{\top, \bot\}$ defined via $\delta_\mathbf{S}(\bot)^{-1} = \{0^\omega\}$, we can then define the represented space $\mathcal{O}(\mathbf{X})$ of *open* subsets of $\mathbf{X}$ by identifying a subset of $\mathbf{X}$ with its (continuous) characteristic function into $\mathbb{S}$. Since countable *or* and binary *and* on $\mathbb{S}$ are computable, so are countable union and binary intersection of open sets.

The space $\mathcal{A}(\mathbf{X})$ of closed subsets is obtained by taking formal complements, i.e. the names for $A \in \mathcal{A}(\mathbf{X})$ are the same as the names of $X \setminus A \in \mathcal{O}(\mathbf{X})$ (i.e. we are using the negative information representation). Intuitively, this means that when reading a name for a closed set, this can always shrink later on, but never grow. It is often very convenient that we can alternatively view $A \in \mathcal{A}(\{0,1\}^\mathbb{N})$ as being represented by some tree T via $[T] = A$ (here $[T]$ denotes the set of infinite paths through T).

We can now define Weihrauch reducibility. Again, we give a very brief treatment here, and refer to [7] for more details and references.

Definition 2 (Weihrauch reducibility). *Let f, g be multivalued functions on represented spaces. Then f is said to be* Weihrauch reducible *to g, in symbols $f \leq_W g$, if there are computable functions $K, H :\subseteq \mathbb{N}^\mathbb{N} \to \mathbb{N}^\mathbb{N}$ such that $(p \mapsto K\langle p, GH(p)\rangle) \vdash f$ for all $G \vdash g$.*

The Weihrauch degrees (i.e. equivalence classes of $\leq_\mathrm{W}$) form a distributive lattice, but we will not need the lattice operations in this paper. Instead, we use two kinds of products. The usual cartesian product induces an operation $\times$ on Weihrauch degrees. We write f^k for the k-fold cartesian product with itself. The compositional product $f \star g$ satisfies that

$$f \star g \equiv_\mathrm{W} \max_{\leq_\mathrm{W}} \{f_1 \circ g_1 \mid f_1 \leq_\mathrm{W} f \wedge g_1 \leq_\mathrm{W} g\}$$

and thus is the hardest problem that can be realized using first g, then something computable, and finally f. The existence of the maximum is shown in [12] via an explicit construction, which is relevant in some proofs. Both products as well as the lattice-join can be interpreted as logical *and*, albeit with very different properties.

We'll briefly mention a further unary operation on Weihrauch degrees, the finite parallelization f^*. This has as input a finite tuple of instances to f and needs to solve all of them.

As mentioned in the introduction, the closed choice principles are valuable benchmark degrees in the Weihrauch lattice:

Definition 3. *For a represented space $\mathbf{X}$, the closed choice principle $C_\mathbf{X} :\subseteq \mathcal{A}(\mathbf{X}) \rightrightarrows \mathbf{X}$ takes as input a non-empty closed subset A of $\mathbf{X}$ and outputs some point $x \in A$.*

3 The principles under investigation

We proceed to give formal definitions of the three problems our investigation is focused on. These are *finite choice*, the task of selecting a point from a closed subset (of $\{0,1\}^{\mathbb{N}}$ or $[0,1]^n$) which is guaranteed to have either exactly or no more than k elements; *convex choice*, the task of selecting a point from a convex closed subset of $[0,1]^k$; and *sorting* an infinite sequence over the alphabet $\{0,1,\ldots,k\}$ in increasing order. Our main result is each task forms a strictly increasing chain in the parameter k, and these chains are perfectly aligned as depicted in Figure 1. For finite choice and convex choice, this was already established in [20]. Our Theorem 5 implies the main theorem from [20] with a very different proof technique.

Definition 4 ([20, Definition 7]). *For a represented space* $\mathbf{X}$ *and* $1 \leq n \in \mathbb{N}$, *let* $C_{\mathbf{X},\sharp=n} := C_{\mathbf{X}}|_{\{A \in \mathcal{A}(\mathbf{X}) \mid |A|=n\}}$ *and* $C_{\mathbf{X},\sharp\leq n} := C_{\mathbf{X}}|_{\{A \in \mathcal{A}(\mathbf{X}) \mid 1 \leq |A| \leq n\}}$.

It was shown as [20, Corollary 10] that for every computably compact computably rich computable metric space $\mathbf{X}$ we find $\mathrm{C}_{\mathbf{X},\sharp=n} \equiv_{\mathrm{W}} \mathrm{C}_{\{0,1\}^{\mathbb{N}},\sharp=n}$ and $\mathrm{C}_{\mathbf{X},\sharp\leq n} \equiv_{\mathrm{W}} \mathrm{C}_{\{0,1\}^{\mathbb{N}},\sharp\leq n}$. This in particular applies to $\mathbf{X} = [0,1]^d$. We denote this Weihrauch degree by $\mathrm{C}_{\sharp=n}$ respectively $\mathrm{C}_{\sharp\leq n}$.

Definition 5 ([20, Definition 8]). *By* XC_n *we denote the restriction of* $C_{[0,1]^n}$ *to convex sets.*

Since for subsets of $[0,1]$ being an interval, being convex and being connected all coincide, we find that XC_1 is the same thing as one-dimensional connected choice CC_1 as studied in [10] and as interval choice $\mathrm{C_I}$ as studied in [3].

Definition 6. *Let* $\mathrm{Sort}_d : d^\omega \to d^\omega$ *be defined by* $\mathrm{Sort}_d(p) = 0^{c_0}1^{c_1}\ldots k^\infty$, *where* $|\{n \mid p(n) = 0\}| = c_0$, $|\{n \mid p(n) = 1\}| = c_1$, *etc, and* k *is the least such that* $|\{n \mid p(n) = k\}| = \infty$. *We write just* Sort *for* Sort_2.

Sort was introduced and studied in [22], and then generalized to Sort_k in [9]. Note that the principle just is about sorting a sequence in order without removing duplicates. In [26] it is shown that $\mathrm{Sort}_{n+1} \equiv_{\mathrm{W}} \mathrm{Sort}^n$; it follows that $\mathrm{Sort}^* \equiv_{\mathrm{W}} \mathrm{Sort}_d^* \equiv_{\mathrm{W}} \coprod_{d\in\mathbb{N}} \mathrm{Sort}_d$. The degree Sort^* was shown in [22] to capture the strength of the strongly analytic machines [13, 15], which in turn are an extension of the BSS-machines [1]. Sort is equivalent to Thomae's function; and to the translation of the standard representation of the reals into the continued fraction representation [28]. In [17], Sort is shown to be equivalent to certain projection operators.

There are some additional Weihrauch problems we make passing reference to. *All-or-unique choice* captures the idea of a problem either having a unique solution, or being completely undetermined:

Definition 7. *Let* $AoUC_{\mathbf{X}}$ *be the restriction of* $C_{\mathbf{X}}$ *to* $\{\{x\} \mid x \in \mathbf{X}\} \cup \{X\}$.

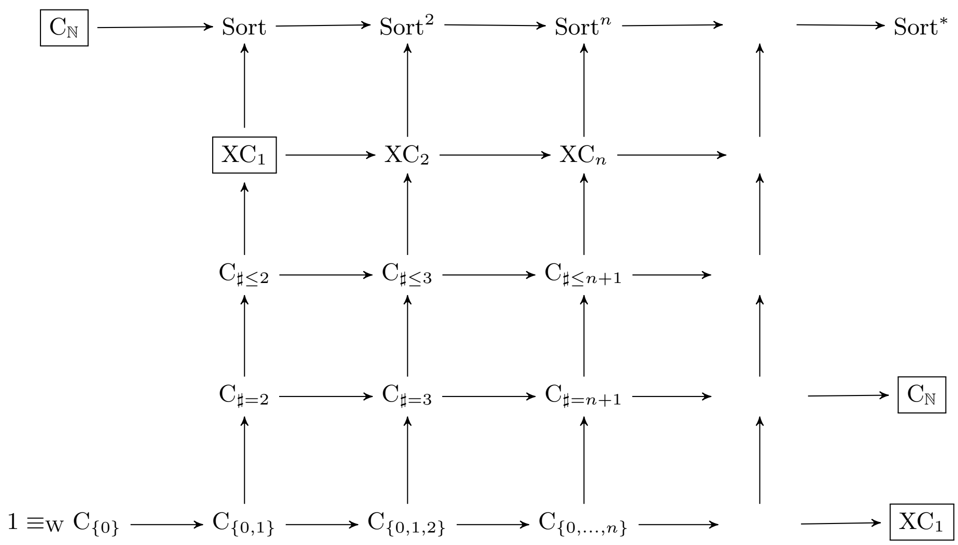

Fig. 1. Overview of our results; extending [20, Figure 1] by the top row. The diagram depicts all Weihrauch reductions between the stated principles up to transitivity. Boxes mark degrees appearing in two places in the diagram. Our additional results are provided as Theorems 3 and 5.

A prototypical example (which is equivalent to the full problem) is solving $ax = b$ over $[0, 1]$ with $0 \leq b \leq a$: Either there is the unique solution $\frac{b}{a}$, or $b = a = 0$, and any $x \in [0, 1]$ will do. The degree of $\mathrm{AoUC}_{\mathbf{X}}$ is the same for any computably compact computably rich computable metric space, in particular for $\mathbf{X} = \{0, 1\}^{\mathbb{N}}$ or $\mathbf{X} = [0, 1]^d$. We just write AoUC for that degree. This problem was studied in [23, 19] where it is shown that AoUC^* is the degree of finding Nash equilibria in bimatrix games and of executing Gaussian elimination.

4 Proving separations via the recursion theorem

A core technique we use to prove our separation results invokes Kleene's recursion theorem in order to let us prove a separation result by proving computability of a certain map (rather than having to show that no computable maps can witness a reduction). We had already used this technique in [19], but without describing it explicitly. Since the technique has proven very useful, we formally state the argument here as Theorem 1 after introducing the necessary concepts to formulate it.

Definition 8. *A representation δ of $\mathbf{X}$ is* precomplete*, if every computable partial $f :\subseteq 2^\omega \to \mathbf{X}$ extends to a computable total $F : 2^\omega \to \mathbf{X}$.*

Proposition 1. *For effectively countably-based $\mathbf{X}$, the space $\mathcal{O}(\mathbf{X})$ (and hence $\mathcal{A}(\mathbf{X})$) is precomplete.*

Proof. It suffices to show this for $\mathcal{O}(\mathbb{N})$, where it just follows from the fact that we can delay providing additional information about a set as long as we want; and will obtain a valid name even if no additional information is forthcoming.

The preceding proposition is a special case of [27, Theorem 6.5], which shows that many pointclasses have precomplete representations.

Proposition 2. *The subspaces of $\mathcal{A}([0,1]^n)$ consisting of the connected respectively the convex subsets are computable multi-valued retracts, and hence precomplete.*

Proof. For the connected sets, this follows from [10, Proposition 3.4]; for convex subsets this follows from computability of the convex hull operation on $[0,1]^n$, see e.g. [20, Proposition 1.5] or [30].

By $\mathcal{M}(\mathbf{X}, \mathbf{Y})$ we denote the represented space of strongly continuous multi-valued functions from $\mathbf{X}$ to $\mathbf{Y}$ studied in [12]. The precise definition of strong continuity is irrelevant for us, we only need every partial continuous function on $\{0,1\}^\mathbb{N}$ induces a minimal strongly continuous multivalued function that it is a realizer of; and conversely, every strongly continuous multivalued function is given by a continuous partial realizer.

Theorem 1. *Let $\mathbf{X}$ have a total precomplete representation. Let $f : \mathbf{X} \rightrightarrows \mathbf{Y}$ and $g : \mathbf{U} \rightrightarrows \mathbf{V}$ be such that there exists a computable $e : \mathbf{U} \times \mathcal{M}(\mathbf{V}, \mathbf{Y}) \rightrightarrows \mathbf{X}$ such that if $x \in e(u,k)$ and $v \in g(u)$, then $k(v) \nsubseteq f(x)$. Then $f \nleq_W g$.*

Proof. Assume that $f \leq_\mathrm{W} g$ via computable H, K. Let computable E be a realizer of e. Let $(\phi_n :\subseteq \mathbb{N} \to \mathbb{N})_{n \in \mathbb{N}}$ be a standard enumeration of the partial computable functions. By assumption, we can consider each ϕ_n to denote some element in $\mathbf{X}$. Let λ be a computable function such that $\phi_{\lambda(n)} = E(H(\phi_n), (v \mapsto K(\phi_n, v)))$. By Kleene's recursion theorem, there is some n_0 with $\phi_{n_0} = \phi_{\lambda(n_0)}$. Inputting ϕ_{n_0} to f fails the assumed reduction witnesses.

As simple sample application for how to prove separations of Weihrauch degrees via the recursion theorem, we shall point out that XC_1 already cannot solve some simple products. For contrast, however, note that $\mathrm{C}_2^* \leq_\mathrm{W} \mathrm{XC}_1$ was shown as [10, Proposition 9.2].

Theorem 2. *$C_2 \times AoUC \nleq_W XC_1$.*

Proof. Given a convex tree $T \subseteq 2^{<\omega}$ and a partial continuous function $\phi :\subseteq \{0,1\}^\mathbb{N} \to 2 \times \{0,1\}^\mathbb{N}$, we compute set $S \in \mathcal{A}(\{0,1\})$ and $V \in \mathcal{A}(\{0,1\}^\mathbb{N})$ such that $S \neq \emptyset$, and $V = \{0,1\}^\mathbb{N}$ or $V = \{p\}$ for some $p \in \{0,1\}^\mathbb{N}$. Our construction ensures that $\exists p \in [T]\ \phi(p) \notin S \times V$.

Initially, $S = \{0,1\}$ and $V = \{0,1\}^\mathbb{N}$.

We first search for s such that for any $\sigma \in T$ of length s, the first value of $\phi(\sigma)$ is determined. If we never find one, then $S = \{0,1\}$ and $V = \{0,1\}^\mathbb{N}$ work as desired.

Next, we search for some $\tau \in \{0,1\}^s$ such that $P_\tau := [T] \cap \bigcup_{j<2} \phi^{-1}(j, [\tau])$ is such that any interval contained in P_τ is contained in some $[\sigma]$ for $\sigma \in \{0,1\}^s$. Note that if $(J_i)_{i \in I}$ is a collection of pairwise disjoint intervals in $\{0,1\}^{\mathbb{N}}$ such that every J_i intersects with at least two cylinders $[\sigma]$ and $[\sigma']$ for some strings $\sigma \neq \sigma'$ of length s, then the size of I is at most $2^s - 1$. Hence, if ϕ is defined on $[T]$, such a τ has to exist. Once we have found it, we set $V = \{\tau 0^\omega\}$.

Either we are already done (since we would have that $\exists p \in [T]\ \phi(p) \notin S \times V$), or it holds that $[T] \subseteq [\sigma]$ for some $\sigma \in \{0,1\}^s$. In that case, by choice of s we find that $\exists j \in \{0,1\}\ \pi_0 \phi(p) = j$ for all $p \in [T]$. We can set $S = \{1-j\}$, and have obtained the desired property that $\exists p \in [T]\ \phi(p) \notin S \times V$. By Theorem 1, the claim follows.

5 Some observations on Sort

5.1 Displacement principle for Sort_k

The basic phenomenon that the number of parallel copies of Sort being used corresponds to a dimensional feature can already by a result similar in feature to the displacement principle from [10]:

Proposition 3. *$C_2 \times f \leq_W \mathrm{Sort}_{k+1}$ implies $f \leq_W \mathrm{Sort}_k \times C_{\mathbb{N}}$.*

Proof. Let the reduction $\mathrm{C}_2 \times f \leq_{\mathrm{W}} \mathrm{Sort}_{k+1}$ be witnessed by computable H, K_1, K_2. Assume, for the sake of a contradiction, that for some input x to f and a name p for $\{0,1\}$ it holds that $H(p,x)$ contains infinitely many 0s. In that case, $\mathrm{Sort}_k(H(p,x)) = 0^\omega$, and hence K_1 is defined as either 0 or 1 on $p, x, 0^\omega$. But then there is some $k \in \mathbb{N}$ such that K_1 already outputs the answer on reading some prefix $p_{\leq k}, x_{\leq k}, 0^k$. Additionally, we can chose some $k' \geq k$ such that H writes at least k' 0s upon reading the prefixes $p_{\leq k'}, x_{\leq k'}$. By changing p after k' to be a name of $\{1 - K_1(p, x, 0^\omega)\}$ shows the contradiction.

Now we note that $x \mapsto H(p,x)$ and K_2 witness a reduction from f to the restriction of Sort_{k+1} to inputs containing only finitely many 0s. But this restriction is reducible to $\mathrm{Sort}_k \times \mathrm{C}_{\mathbb{N}}$: In parallel, call Sort_k on the sequence obtained by skipping 0s and decrementing every other digit by 1, and using $\mathrm{C}_{\mathbb{N}}$ to determine the original number of 0s.

Corollary 1. *Let f be a closed fractal. Then $C_2 \times f \leq_W \mathrm{Sort}_{k+1}$ implies $f \leq_W \mathrm{Sort}_k$.*

Corollary 2. $C_2 \times C^n_{\sharp \leq 2} \nleq_W \mathrm{Sort}_{n+1}$.

Corollary 3. $C_2 \times XC^n_1 \nleq_W \mathrm{Sort}_{n+1}$

We also get an alternative proof of the following, which was previously shown in [22] using the squashing principle from [14]:

Corollary 4. $\mathrm{Sort}_{k+1} \nleq_W \mathrm{Sort}_k$

5.2 Sort and convex choice

The one-dimensional case of the following theorem was already proven as [9, Proposition 16]:

Theorem 3. $XC_n \leq_W \mathrm{Sort}_{n+1}$

Proof. Let $(H_i^d)_{i \in \mathbb{N}}$ be an effective enumeration of the d-dimensional rational hyperplanes for each $d \leq n-1$. Given $A \in \mathcal{A}([0,1]^n)$, we can recognize that $A \cap H_i^d = \emptyset$ by compactness of $[0,1]^n$. We proceed to compute an input p to Sort_{n+1} as follows:

We work in stages $(\ell_0, \ldots, \ell_{n-1})$. We simultaneously test whether $A \cap H_{\ell_0}^{n-1} = \emptyset$, whether $A \cap H_{\ell_0}^{n-1} \cap H_{\ell_1}^{n-2} = \emptyset$, ..., and whether $A \cap H_{\ell_0}^{n-1} \cap \ldots \cap H_{\ell_{n-1}}^{1} = \emptyset$.

If we find a confirmation for a query involving ℓ_k as the largest index, we write a k to p, increment ℓ_k by 1, and reset any ℓ_i for $i > k$. All tests of smaller indices are continued (and hence will eventually fire if true before a largest index test interferes). In addition, we write ns to p all the time to ensure an infinite result.

Now consider the output $\mathrm{Sort}_{n+1}(p)$. If this is 0^ω, then A does not intersect any $n-1$-dimensional rational hyperplane at all. As a convex set, A has to be a singleton. Thus, as long as we read 0s from $\mathrm{Sort}_{n+1}(p)$, we can just wait until A shrinks sufficiently to produce the next output approximation. If we ever read a 1 in $\mathrm{Sort}_{n+1}(p)$ at position t, we have thus found a $n-1$-dimensional hyperplane H_t^{n-1} intersecting A. We can compute $A \cap H_t^{n-1} \in \mathcal{A}([0,1]^n)$, and proceed to work with that set. By retracing the computation leading up to the observation that $A \cap H_{t-1}^{n-1} = \emptyset$, we can find out how many larger-index tests were successful before that. We disregard their impact on $\mathrm{Sort}_{n+1}(p)$. Now as long as we keep reading 1s, we know that $A \cap H_{t-1}^{n-1}$ is not intersecting $n-2$-dimensional rational hyperplanes (and hence could be singleton). Finding a 2 means we have identified a $n-2$-dimensional hyperplane $H_{t'}^{n-2}$ intersecting $A \cap H_{t-1}^{n-1}$, and we proceed to work with $A \cap H_{t-1}^{n-1} \cap H_{t'}^{n-2}$. Continuing this process, we always find that either our set has been collapsed to singleton (from which we can extract the point), or we will be able to reduce its dimension further (which can happen only finitely many times).

5.3 A digression: Sort and finding connected components of a graph

On a side note, we explore how Sort relate to the problem FCC of finding a connected component of a countable graph with only finitely many connected components. Here the graph (V, E) is given via the characteristic functions of $V \subseteq \mathbb{N}$ and $E \subseteq \mathbb{N} \times \mathbb{N}$, and the connected component is to be produced likewise as its characteristic function. In addition, we have available to us an upper bound for the number of its connected components. In the reverse math context, this problem was studied in [18] and shown to be equivalent to Σ_2^0-induction.

Theorem 4. *The following are equivalent:*

1. *FCC*
2. Sort^*

Proof. $\mathbf{FCC} \leq_{\mathbf{W}} \coprod_{k \in \mathbb{N}} \text{Sort}_k$

We are given $n \in \mathbb{N}$ and a graph with at most n connected components. For each $2 \leq i \leq n$, we pick some standard enumeration $(V_j^i)_{j \in \mathbb{N}}$ of the i-element subsets of $\mathbb{N}$. As soon as we learn that none of the V_j^i with $j \leq l$ is an independent set, we write the l-th symbol $i-2$ on the input to Sort_{n-1}. We write an $n-1$ occasionally to ensure that the output is actually infinite. Now assume we have access to the corresponding output q of Sort_{n-1}. This will be 0^ω iff the graph had a single connectedness component, and of the form $0^l 1 p$ else where V_l^2 is an independent pair. We can thus start computing the connectedness component of 0 by searching in parallel whether $q \neq 0^\omega$ and searching for a path from 0 to the current number. Either search will terminate. In the latter case, we can answer yes. In the former, we now search for paths to the two vertices in the pair (and thus might be answer to correctly no). Simultaneously we investigate the remnant p whether $p = 1^\omega$ (and thus the graph has 2 connectedness components, and any vertex is linked to either member of V_l^2), or find an independent set of size 3, etc.

$\text{Sort}_k \equiv_{\mathbf{W}} \text{Sort}^{k-1}$

This was shown in [26].

$\text{Sort} \leq_{\mathbf{W}}$ **FCC**

We compute a graph with at most 2 connectedness components. The graph will be bipartite, with the odd and even numbers being separate components. All odd numbers are connected to 0, and at any stage there will be some even number $2n$ not yet connected to 0, which represents some number i such that we have not yet read i times 0 in the input p to Sort. If we read the i-th 0 in p at time t, we connect $2t+1$ to both 0 and $2n$. If we read a 1 at time t, then $2t+1$ gets connected to 0 and $2t$.

If p contains infinitely many 0s, then we end up with a single connectedness component. Otherwise we obtain either the connectedness component of 0, or equivalently, its complement. Once we see that e.g. 2 is in this connectedness component, then we can output 0. Moreover, then 2 must be linked to 0 via some $2t+1$ (which we can exhaustively search for), and whether $2t$ is in the connectedness component tells us whether the next bit of the output is 1 (and then continuous as 1^ω), or 0 again, in which case we need to search for the next significant digit.

$\mathbf{FCC} \times \mathbf{FCC} \leq_{\mathbf{W}} \mathbf{FCC}$

Just use the product graph.

6 Finite choice and sorting

Theorem 5. $\mathsf{C}_{\# \leq k+1} \not\leq_{\mathsf{W}} \mathsf{Sort}_k$.

Proof. By the recursion theorem, it suffices to describe an effective procedure which, given $\alpha \in k^\omega$ and Φ, constructs an instance C of $\mathsf{C}_{\# \leq k+1}$ such that there is a solution q to $\mathsf{Sort}_k(\alpha)$ such that $\Phi(q)$ is not a solution to C.

For a finite tree T of height s, we say that $\sigma \in T$ is *extendible* if there is a leaf $\rho \in T$ of height s which extends σ. Note that an instance of $\mathsf{C}_{\# \leq k+1}$ is generated by an increasing sequence $(T_s)_{s\in\omega}$ of finite binary trees satisfying the following conditions for every s.

(I) T_s is of height s, and T_s has at least one, and at most $k+1$ extendible leaves.
(II) Every node $\sigma \in T_{s+1} \setminus T_s$ is of length $s+1$, and extends an extendible leaf of T_s.

More precisely, for such a sequence (T_s), the union $T = \bigcup_s T_s$ forms a (T_s)-computable tree which has at most $k+1$ many infinite paths. Therefore, the set of all infinite paths through T is an instance of $\mathsf{C}_{\# \leq k+1}$.

For $\eta \in k^{<\omega}$ and $u < k$, let $N[\eta, u]$ be the number of the occurrences of u's in η, i.e., $N[\eta, u] = \#\{i : \eta(i) = u\}$. We define the *$u$-partial sort of* η as the following string:

$$(\eta)_u^{\mathrm{sort}} = 0^{N[\eta,0]}1^{N[\eta,1]}2^{N[\eta,2]} \dots (u-1)^{N[\eta,u-1]}.$$

Our description of an effective procedure which, given an instance α of Sort_k, returns a sequence $(T_s)_{s\in\omega}$ of finite trees generating an instance of $\mathsf{C}_{\# \leq k+1}$ is subdivided into k many strategies $(\mathcal{S}_u)_{u<k}$. At stage s, the u-th strategy $\mathcal{S}_u$ for $u < k$ believes that u is the least number occurring infinitely often in a given instance α of Sort_k, and there is no $i \geq s$ such that $\alpha(i) < u$. In other words, the strategy $\mathcal{S}_u$ believes that $(\alpha \upharpoonright s)_u^{\mathrm{sort}\frown} u^\omega$, the u-partial sort of the current approximation of α followed by the infinite constant sequence u^ω, is the right answer to the instance α of Sort_k. Then, the strategy $\mathcal{S}_u$ waits for $\Phi((\alpha \upharpoonright s)_u^{\mathrm{sort}\frown} u^\omega)$ being a sufficiently long extendible node ρ of T_s, and then make a branch immediately after an extendible leaf $\rho_u \in T_s$ extending ρ, where this branch will be used for diagonalizing $\Phi((\alpha \upharpoonright s)_u^{\mathrm{sort}\frown} u^\omega)$. This action injures all lower priority strategies $(\mathcal{S}_v)_{u<v<k}$ by initializing their states and letting ρ_v be undefined.

More precisely, each strategy $\mathcal{S}_u$ has a state, $\mathtt{state}_s(u) \in \{0,1,2\}$, at each stage s, which is initialized as $\mathtt{state}_0(u) = 0$. We also define a partial function $u \mapsto \rho_u^s$ for each s, where ρ_u^s is extendible in T_s if it is defined. Roughly speaking, ρ_u^s is the stage s approximation of the diagonalize location for the u-th strategy as described above. We assume that ρ_u^0 is undefined for $u > 0$, for any $s \in \omega$, ρ_0^s is defined as an empty string, and ρ_u^s is a finite string whenever it is defined.

At the beginning of stage $s+1$, inductively assume that a finite tree T_s of height s and a partial function $u \mapsto \rho_u^s$ has already been defined. Moreover, we inductively assume that if $\mathtt{state}_s(u) = 1$ then ρ_u^s is defined, and $\rho_u^{s\frown} i$ is extendible in T_s for each $i < 2$. At substage u of stage $s+1$, the strategy $\mathcal{S}_u$ acts as follows:

1. If $(\alpha \upharpoonright s+1)_u^{\mathrm{sort}} \neq (\alpha \upharpoonright s)_u^{\mathrm{sort}}$, then initialize the strategy, that is, put $\mathtt{state}_{s+1}(u) = 0$, and let ρ_u^{s+1} be undefined. Then go to the next substage $u+1$ if $u < k$; otherwise go to the next stage $s+2$.

2. If $(\alpha \upharpoonright s+1)^{\rm sort}_u = (\alpha \upharpoonright s)^{\rm sort}_u$ and $\mathtt{state}_s(u) = 0$, then ask if $\Phi((\alpha \upharpoonright s)^{\rm sort}_u {}^\frown u^\omega)[s]$ is an extendible node $\rho \in T_s$ such that for any $v < u$, if ρ^s_v is defined, then $\rho \not\preceq \rho^s_v$ holds.
   (a) If yes, define ρ^{s+1}_u as the leftmost extendible leaf of T_s extending such a ρ, and put $\mathtt{state}_{s+1}(u) = 1$. Injure all lower priority strategies, that is, put $\mathtt{state}_{s+1}(v) = 0$ and let ρ^{s+1}_v be undefined for any $u < v < k$. Then go to the next stage $s+2$.
   (b) If no, go to the next substage $u+1$ if $u < k$; otherwise go to the next stage $s+2$.
3. If $(\alpha \upharpoonright s+1)^{\rm sort}_u = (\alpha \upharpoonright s)^{\rm sort}_u$ and $\mathtt{state}_s(u) = 1$, then ask if $\Phi((\alpha \upharpoonright s)^{\rm sort}_u {}^\frown u^\omega)[s]$ is an extendible node $\rho \in T_s$ which extends $\rho^s_u {}^\frown i$ for some $i < 2$.
   (a) If yes, define $\rho^{s+1}_u = \rho^s_u {}^\frown (1-i)$ for such i, and put $\mathtt{state}_{s+1}(u) = 2$. Injure all lower priority strategies, that is, put $\mathtt{state}_{s+1}(v) = 0$ and let ρ^{s+1}_v be undefined for any $u < v < k$. Then go to the next stage $s+2$.
   (b) If no, go to the next substage $u+1$ if $u < k$; otherwise go to the next stage $s+2$.
4. If not mentioned, set $\mathtt{state}_{s+1}(u) = \mathtt{state}_s(u)$ and $\rho^{s+1}_u = \rho^s_u$.

At the end of stage $s+1$, we will define T_{s+1}. Consider the downward closure T^*_{s+1} of the following set:

$$\{\rho^{s+1}_u {}^\frown i : \mathtt{state}(u) = 1 \text{ and } i < 2\} \cup \{\rho^{s+1}_u : \mathtt{state}(u) = 2\}.$$

Let $T^{*,\rm leaf}_{s+1}$ be the set of all leaves of T^*_{s+1}. Note that every element of $T^{*,\rm leaf}_{s+1}$ is extendible in T_s since ρ^{s+1}_u is extendible in T_s. For each leaf $\rho \in T^{*,\rm leaf}_{s+1}$, if $|\rho| = s+1$ then put $\eta_\rho = \eta$; otherwise choose an extendible leaf $\eta \in T_s$ extending ρ, and define $\eta_\rho = \eta^\frown 0$.

Let T_0 be an empty tree. We define T_{s+1} as follows:

$$T_{s+1} = T_s \cup \{\eta_\rho : \rho \in T^{*,\rm leaf}_{s+1}\}.$$

Note that the extendible nodes in T_{s+1} are exactly the downward closure of $\{\eta_\rho : \rho \in T^{*,\rm leaf}_{s+1}\}$, and every element of T^*_{s+1} is extendible in T_{s+1}, that is,

– If $\mathtt{state}_{s+1}(u) = 1$, then $\rho^{s+1}_u {}^\frown i$ is extendible in T_{s+1} for each $i < 2$.
– If $\mathtt{state}_{s+1}(u) = 2$, then ρ^{s+1}_u is extendible in T_{s+1}.

Our definition of $(T_s)_{s \in \omega}$ clearly satisfies the property (II) mentioned above. Concerning the property (I), one can see the following:

Lemma 1. *T_{s+1} has at least one, and at most $k+1$ extendible leaves.*

Proof. The former assertion trivially holds since ρ^s_0 is always defined as an empty string for any $s \in \omega^\omega$. For the latter assertion, it suffices to show that any branching extendible node of T_{s+1} is of the form ρ^{s+1}_u for some $u < k$. This is because T_s is binary, and then the above property automatically ensures that T_s has at most $k+1$ extendible leaves.

Let σ be a branching extendible node of T_{s+1}. If $|\sigma| = s$, since T_s is of height s, σ is of the form ρ_u^{s+1} by our definition of T_{s+1} . If $|\sigma| < s$, then it is also a branching extendible node of T_s by the property (II) of our construction, and thus it is of the form ρ_u^s by induction. If $\rho_u^s = \rho_u^{s+1}$ for any u, then our Lemma clearly holds. If $\rho_u^s \neq \rho_u^{s+1}$, then it can happen at (2a) or (3a), and thus, there is $v \leq u$ such that the v-th strategy has acted at stage $s+1$. We claim that for any $\rho \in T^*_{s+1}$ we have $\rho_u^s \not\prec \rho$. This claim implies that ρ_u^s is not a branching extendible node in T_{s+1}, which is a contradiction, and therefore we must have $\rho_u^s = \rho_u^{s+1}$.

To show the claim, note that ρ_w^{s+1} is undefined for $w > v$. If $w < u$ and ρ_w^s is defined then $\rho_u^s \not\preceq \rho_w^s$ by $\mathcal{S}_u$'s action at (2a). If $w < v$ then $\rho_w^{s+1} = \rho_w^s$. For $w = v$, if $\mathtt{state}_{s+1}(v) = 1$ then $\mathcal{S}_v$ reaches at (2a) at stage $s+1$ and $\rho_v^s \not\preceq \rho_u^s$ by $\mathcal{S}_v$'s action. If $\mathtt{state}_{s+1}(v) = 2$ then $\mathcal{S}_v$ reaches at (3a) at stage $s+1$, and thus ρ_v^{s+1} is a successor of ρ_u^{s+1} and thus $\rho_u^s \not\prec \rho_v^{s+1}$. Hence, there is no $\rho \in T^*_{s+1}$ such that $\rho_u^s \prec \rho$ as desired.

Lemma 2. *If* $\mathtt{state}_{s+1}(u) = 2$*, then* $\Phi((\alpha \upharpoonright s+1)_u^{\mathrm{sort}\frown} u^\omega)$ *is not extendible in* T_{s+1}.

Proof. If $\mathtt{state}_{s+1}(u) = 2$, then there is stage $t \leq s+1$ such that $(\alpha \upharpoonright t)_u^{\mathrm{sort}} = (\alpha \upharpoonright s+1)_u^{\mathrm{sort}}$ and the u-th strategy $\mathcal{S}_u$ arrives at (2a) at stage s and (3a) at $s+1$, and the u-th strategy is not injured by any higher priority strategy during stages between t and $s+1$, and in particular, $\rho_u^t = \rho_u^s$. By our action (3a), $\Phi((\alpha \upharpoonright s+1)_u^{\mathrm{sort}\frown} u^\omega)$ extends the sister of ρ_u^{s+1}. If $v > u$ then ρ_v^{s+1} is undefined. If $v < u$ and ρ_v^t is undefined, then since no injury happens below u during stages between t and $s+1$, we have $\rho_u^s = \rho_u^t \not\preceq \rho_v^t = \rho_v^{s+1}$, which implies that ρ_v^{s+1} does not extend the sister of ρ_u^{s+1}. Hence the sister of ρ_u^{s+1} does not extend to a leaf of T^*_{s+1}. Therefore, $\Phi((\alpha \upharpoonright s+1)_u^{\mathrm{sort}\frown} u^\omega)$ is not extendible in T_{s+1}.

We now verify our construction. Put $T = \bigcup_k T_k$. By Lemma 1, since our construction of $(T_s)_{s\in\omega}$ satisfies the conditions (I) and (II), the set $[T]$ of all infinite paths through T is an instance of $\mathsf{C}_{\#\leq k+1}$. Let α be an instance of Sort_k.

Lemma 3. $\Phi(\mathsf{Sort}_k(\alpha)) \notin [T]$.

Proof. By pigeonhole principle, there exists u such that $\alpha(i) = u$ for infinitely many i. Let u be the least such number. Then there exists s such that $(\alpha)^{\mathrm{sort}} := (\alpha \upharpoonright s)_u^{\mathrm{sort}\frown} u^\omega$ is the right answer to the instance α of Sort_k, that is, it is the result by sorting α. Then, for any $v \leq u$, the v-partial sort of α stabilizes after s, that is, $(\alpha \upharpoonright t+1)_v^{\mathrm{sort}} = (\alpha \upharpoonright t)_v^{\mathrm{sort}}$ for all $t \geq s$. After the v-partial sort of α stabilizes, the v-th strategy $\mathcal{S}_v$ can injure lower priority strategies at most two times, i.e., at (2a) and (3a). Therefore, there is stage $s_0 \geq s$ such that the u-th strategy $\mathcal{S}_u$ is never injured by higher priority strategies after s_0. Then, $\mathtt{state}_t(u)$ converges to some value.

Case 1. $\lim_t \mathtt{state}_t(u) = 0$. By our choice of s_0, $\mathcal{S}_u$ always goes to (2b), and never goes to (2a) after s_0. However, if $\Phi((\alpha)^{\mathrm{sort}})$ is an infinite string, then the strategy must go to (2a) since $\{\rho_v^s : v < u\}$ is finite. Hence, $\Phi((\alpha)^{\mathrm{sort}})$ cannot be an infinite path through T.

Case 2. $\lim_t \mathtt{state}_t(u) = 1$. Let $s_1 \geq s_0$ be the least stage such that $\mathcal{S}_u$ reaches (2a) with some ρ. We claim that if an extendible node in T_t extends ρ, then it also extends ρ_u^t for any $t > s_1$. According to the condition of $\mathcal{S}_u$'s strategy (2), for any $v < u$, we have $\rho \not\preceq \rho_v^{s_1} = \rho_v^{s_0}$. By injury in (2a), $\rho_v^{s_1}$ is undefined for any $v > u$. Therefore, any extendible node of T_{s_1+1} extends ρ_v^t or $\rho_v^t \frown i$ for some $v \leq u$ and $i < 2$. Hence, if an extendible node in T_{s_1+1} extends ρ, then it also extends $\rho_u^{s_1+1} = \rho_u^t$. By the property (II) of our construction, the claim follows. Now, by our assumption, $\mathcal{S}_u$ always goes to (3b), and never goes to (3a). This means that $\Phi((\alpha \upharpoonright t)_u^{\mathrm{sort}} \frown u^\omega)$ extends ρ, but does not extend ρ_u^t for any $t > s_1$. Therefore, $\Phi((\alpha \upharpoonright t)_u^{\mathrm{sort}} \frown u^\omega)$ is not extendible in T_t for any $t > s_1$. Consequently, $\Phi((\alpha)^{\mathrm{sort}}) \notin [T]$.

Case 3. $\lim_t \mathtt{state}_t(u) = 2$. Let $s_2 \geq s_0$ be the least stage such that $\mathcal{S}_u$ reaches (3a). Then by Lemma 2, $\Phi((\alpha \upharpoonright s_2)_u^{\mathrm{sort}} \frown u^\omega)$ is not extendible in T_{s_2}. Since $\mathcal{S}_u$ is not injured after s_0, we conclude $\Phi((\alpha)^{\mathrm{sort}}) \notin [T]$.

By the recursion theorem, this obviously implies the desired assertion.

7 The comparison game for products of finite choice

In this section we consider the question when finite choice for some cardinality is reducible to some finite product of finite choice operators. We do not obtain an explicit characterization, but rather an indirect one. We introduce a special reachability game (played on a finite graph), and show that the winner of this game tells us whether the reduction holds. This in particular gives us a decision procedure (which so far has not been implemented yet, though).

Our game is parameterized by numbers k, and $n_0, n_1, \ldots, n_\ell$. We call the elements of $\bigcup_{i \leq \ell} \{i\} \times n_i$ *colours*, and the elements of $\Pi_{i \leq \ell} n_i$ *tokens*. A token w has colour (i, c), if $w_i = c$.

The current board consists of up to k boxes each of which contains some set of tokens, with no token appearing in distinct boxes. If there ever is an empty box, then Player 1 wins. If the game continues indefinitely without a box becoming empty, Player 2 wins. The initial configuration is chosen by Player 1 selecting the number of boxes, and by Player 2 distributing all tokens into these boxes.

The available actions are as follows:

- **Remove** Player 1 taps a box b. Player 2 selects some colours C such that every token in b has a colour from C. Then the box b and all tokens with a colour from C are removed.
- **Reintroduce colour** Player 2 picks two 'adjacent' colours (i, c) and (i, d), such that no token on the board has colour (i, d). For every box b, and every token $w \in b$ having colour c, he then adds a token w' to b that is identical to w except for having colour (i, d) rather than (i, c).

Split box If there are less than k boxes on the board, Player 1 can select a box b to be split into two boxes b_0 and b_1. Player 2 can chose how to distribute the tokens from b between b_0 and b_1. Moreover, Player 2 can do any number of *Reintroduce colour* moves before the *Split box*-move takes effect.

Theorem 6. *$C_{\sharp\leq k} \leq_W C_{\sharp\leq n_0} \times \ldots C_{\sharp\leq n_\ell}$ iff Player 2 wins the comparison game for parameters k, $n_0, \ldots, n_\ell$.*

The proof proceeds via Lemmas 4, 5 below. We observe that the game is a reachability game played on a finite graph. In particular, it is decidable who wins the game for a given choice of parameters. We have only considered the case $n_i = 2$ so far, and know:

Proposition 4.

1. *Player 2 wins for $k + 1 \leq \ell$.*
2. *Player 1 wins for $k + 1 \geq 2^{\ell-1}$*

Proof. The first claim follows from Theorem 6 in conjunction with [20, Proposition 3.9] stating that $\mathrm{C}_{\sharp\leq n+1} \leq_{\mathrm{W}} \mathrm{C}^n_{\sharp\leq 2}$. The second is immediate when analyzing the game.

Lemma 4. *From a winning strategy of Player 2 in the comparison game we can extract witnesses for the reduction $C_{\sharp\leq k} \leq_W C_{\sharp\leq n_0} \times \ldots C_{\sharp\leq n_\ell}$.*

Proof. We recall that the input to $\mathrm{C}_{\sharp\leq k}$ can be seen as an infinite binary tree having at most k vertices on each level. We view this tree as specifying a strategy for Player 1 in the comparison game: The boxes correspond to the paths existing up to the current level of the tree. If a path dies out, Player 1 taps the corresponding box. If a path splits into two, Player 1 splits the corresponding box.

Which tokens exist at a certain time tells us how the instances to $\mathrm{C}_{\sharp\leq n_0}, \ldots, \mathrm{C}_{\sharp\leq n_\ell}$ are built. The colour (i, j) refers to the j-path through the i-th tree at the current approximation. If a colour gets removed, this means that the corresponding path dies out. If a colour gets reintroduced, we split the path corresponding to the duplicated colour into two.

It remains to see how the outer reduction witness maps infinite paths through these trees back to an infinite path through the input tree. If we are currently looking at some finite approximation of the input tree and the query trees, together with an infinite path through each query tree, then the infinite paths indicates some token which never will be removed. That means that any box containing that token never gets tapped, i.e. that certain prefixes indeed can be continued to an infinite path.

Lemma 5. *From a winning strategy of Player 1 in the comparison game we can extract a witness for the non-reduction $C_{\sharp\leq k} \nleq_W C_{\sharp\leq n_0} \times \ldots C_{\sharp\leq n_\ell}$ according to Theorem 1.*

Proof. We need to describe a procedure that constructs an input for $C_{\sharp \leq k}$ given inputs to $C_{\sharp \leq n_0}, \ldots, C_{\sharp \leq n_\ell}$ and an outer reduction witness. Inverting the procedure from Lemma 4, we can view the given objects as describing a strategy of Player 2 in the game. We obtain the input tree to $C_{\sharp \leq k}$ by observing how the winning strategy of Player 1 acts against this. When Player 1 taps the i-th box, we let the i-th path through the tree die out. When Player 1 splits the i-th box, we let both children of the i-th vertex present at the current layer be present at the subsequent layer. Otherwise, we keep the left-most child of any vertex on the previous layer.

Since Player 1 is winning, we will eventually reach an empty box. At that point, we let all other paths die out, and only keep the one corresponding to the empty box. This means that any path selected by the outer reduction witness we obtained Player 2's strategy from will fall outside the tree, and thus satisfy the criterion of Theorem 1.

References

1. Blum, L., Cucker, F., Shub, M., Smale, S.: Complexity and Real Computation. Springer (1998)
2. Brattka, V., de Brecht, M., Pauly, A.: Closed choice and a uniform low basis theorem. Annals of Pure and Applied Logic **163**(8), 968–1008 (2012). https://doi.org/10.1016/j.apal.2011.12.020
3. Brattka, V., Gherardi, G.: Effective choice and boundedness principles in computable analysis. Bulletin of Symbolic Logic **17**, 73 – 117 (2011). https://doi.org/10.2178/bsl/1294186663, arXiv:0905.4685
4. Brattka, V., Gherardi, G.: Weihrauch degrees, omniscience principles and weak computability. Journal of Symbolic Logic **76**, 143 – 176 (2011), arXiv:0905.4679
5. Brattka, V., Gherardi, G., Hölzl, R.: Probabilistic computability and choice. Information and Computation **242**, 249 – 286 (2015). https://doi.org/10.1016/j.ic.2015.03.005, http://arxiv.org/abs/1312.7305
6. Brattka, V., Gherardi, G., Hölzl, R., Pauly, A.: The Vitali Covering Theorem in the Weihrauch Lattice, pp. 188–200. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-50062-1_14, http://dx.doi.org/10.1007/978-3-319-50062-1_14
7. Brattka, V., Gherardi, G., Pauly, A.: Weihrauch complexity in computable analysis. arXiv 1707.03202 (2017)
8. Brattka, V., Hertling, P., Weihrauch, K.: A tutorial on computable analysis. In: Cooper, B., Löwe, B., Sorbi, A. (eds.) New Computational Paradigms: Changing Conceptions of What is Computable. pp. 425–491. Springer (2008)
9. Brattka, V., Hölzl, R., Kuyper, R.: Monte Carlo Computability. In: Vollmer, H., Vallée, B. (eds.) 34th Symposium on Theoretical Aspects of Computer Science (STACS 2017). Leibniz International Proceedings in Informatics (LIPIcs), vol. 66, pp. 17:1–17:14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2017). https://doi.org/10.4230/LIPIcs.STACS.2017.17, http://drops.dagstuhl.de/opus/volltexte/2017/7016
10. Brattka, V., Miller, J., Le Roux, S., Pauly, A.: Connected choice and Brouwer's fixed point theorem. Journal for Mathematical Logic (20XX), accepted for publication, 1206.4809

11. Brattka, V., Pauly, A.: Computation with advice. Electronic Proceedings in Theoretical Computer Science **24** (2010), http://arxiv.org/html/1006.0551, cCA 2010
12. Brattka, V., Pauly, A.: On the algebraic structure of Weihrauch degrees. Logical Methods in Computer Science **14**(4) (2018). https://doi.org/10.23638/LMCS-14(4:4)2018, http://arxiv.org/abs/1604.08348
13. Chadzelek, T., Hotz, G.: Analytic machines. Theoretical Computer Science **219**, 151–167 (1999)
14. Dorais, F.G., Dzhafarov, D.D., Hirst, J.L., Mileti, J.R., Shafer, P.: On uniform relationships between combinatorial problems. Transactions of the AMS **368**, 1321–1359 (2016). https://doi.org/10.1090/tran/6465, arXiv 1212.0157
15. Gärtner, T., Hotz, G.: Computability of analytic functions with analytic machines. In: Ambos-Spies, K., Löwe, B., Merkle, W. (eds.) Mathematical Theory and Computational Practice, Lecture Notes in Computer Science, vol. 5635, pp. 250–259. Springer (2009). https://doi.org/10.1007/978-3-642-03073-4_26, http://dx.doi.org/10.1007/978-3-642-03073-4_26
16. Gherardi, G., Marcone, A.: How incomputable is the separable Hahn-Banach theorem? Notre Dame Journal of Formal Logic **50**(4), 393–425 (2009). https://doi.org/10.1215/00294527-2009-018
17. Gherardi, G., Marcone, A., Pauly, A.: Projection operators in the weihrauch lattice. Computability (20XX), accepted for publication, arXiv 1805.12026
18. Gura, K., Hirst, J.L., Mummert, C.: On the existence of a connected component of a graph. Computability **4**(2), 103–117 (2015). https://doi.org/10.3233/COM-150039
19. Kihara, T., Pauly, A.: Dividing by Zero – How Bad Is It, Really? In: Faliszewski, P., Muscholl, A., Niedermeier, R. (eds.) 41st Int. Sym. on Mathematical Foundations of Computer Science (MFCS 2016). Leibniz International Proceedings in Informatics (LIPIcs), vol. 58, pp. 58:1–58:14. Schloss Dagstuhl (2016). https://doi.org/10.4230/LIPIcs.MFCS.2016.58
20. Le Roux, S., Pauly, A.: Finite choice, convex choice and finding roots. Logical Methods in Computer Science (2015). https://doi.org/10.2168/LMCS-11(4:6)2015, http://arxiv.org/abs/1302.0380
21. Neumann, E.: Computational problems in metric fixed point theory and their Weihrauch degrees. Logical Methods in Computer Science **11**(4) (2015). https://doi.org/10.2168/LMCS-11(4:20)2015
22. Neumann, E., Pauly, A.: A topological view on algebraic computations models. Journal of Complexity **44** (2018). https://doi.org/10.1016/j.jco.2017.08.003, http://arxiv.org/abs/1602.08004
23. Pauly, A.: How incomputable is finding Nash equilibria? Journal of Universal Computer Science **16**(18), 2686–2710 (2010). https://doi.org/10.3217/jucs-016-18-2686
24. Pauly, A.: On the (semi)lattices induced by continuous reducibilities. Mathematical Logic Quarterly **56**(5), 488–502 (2010). https://doi.org/10.1002/malq.200910104
25. Pauly, A.: On the topological aspects of the theory of represented spaces. Computability **5**(2), 159–180 (2016). https://doi.org/10.3233/COM-150049, http://arxiv.org/abs/1204.3763
26. Pauly, A., Tsuiki, H.: T^ω-representations of compact sets. arXiv:1604.00258 (2016)
27. Selivanov, V.L.: Total representations. Logical Methods in Computer Science **9**(2) (2013)
28. Weihrauch, K.: The degrees of discontinuity of some translators between representations of the real numbers. Informatik Berichte 129, FernUniversität Hagen, Hagen (Jul 1992)

29. Weihrauch, K.: Computable Analysis. Springer-Verlag (2000)
30. Ziegler, M.: Computable operators on regular sets. Mathematical Logic Quarterly **50**, 392–404 (2004)

Acknowledgement

We are grateful to Stéphane Le Roux for a fruitful discussion leading up to Theorems 2 and 3.