Highness for isomorphism at Scott rank ω_1^{CK}

Abstract

We prove that a Turing degree is high for Scott rank ω_1^{CK} if and only if it is high for isomorphism, answering Question 1 of Franklin, Rossegger, and Turetsky (arXiv, 2026). The result follows from a uniform encoding of arbitrary computable tree path problems by isomorphism problems between computable structures of Scott rank at most ω_1^{CK} in a fixed finite language. The structures are isomorphic exactly when the tree has an infinite path, and paths and isomorphisms are uniformly computable from one another. The key step is a normal form preserving the path problem in which the extendible nodes are computable at each fixed level, with the decision procedure allowed to depend nonuniformly on the level. We obtain this form by coupling a binary path code with numerical decoding bounds: upward closure and Dickson's lemma give a finite description of extendibility at each level. Applying the classical Boolean group-tree construction then yields computable infinitary definitions of all finite tuple orbits. The construction and both reductions are uniform, while the highness equivalence uses the usual nonuniform convention.

Note. This paper was generated entirely by AI, including MiMo, using an automated research pipeline developed by Chenghua Liu and Hanyu Li.

1 Introduction

A Turing degree is *high for isomorphism* if it computes an isomorphism between every pair of isomorphic computable structures. This notion, introduced by Calvert, Franklin, and Turetsky [1], measures the strength needed to identify computable presentations of the same structure. Restricting the structures by Scott rank relates this computational problem to the complexity of defining automorphism orbits.

Write $\kappa = \omega_1^{\mathrm{CK}}$. Franklin, Rossegger, and Turetsky [4] ask whether a degree can be high for Scott rank κ without being high for Scott rank $\kappa+1$, and hence without being high for isomorphism. Their Theorem 19 shows that highness for rank κ suffices to compute a descending sequence in every ill-founded computable linear order. We prove the stronger conclusion that it suffices to compute a path through every ill-founded computable tree. By [1, Proposition 2.6], this is equivalent to highness for isomorphism and therefore answers their Question 1 negatively.

Computable structures of rank κ already have a substantial construction theory. Makkai [6] produced an arithmetical example, Knight and Millar [5] obtained computable examples, and Calvert, Knight, and Millar [2] realized the rank by computable trees. The group-tree construction presented in [5, §3] encodes paths by automorphisms. Its Scott-rank analysis uses computable bounds on the well-founded node ranks at each fixed level [5, Theorem 3.7]. The issue here is to meet such a local rank condition while preserving the path problem of an arbitrary input tree, rather than constructing a particular example of high Scott rank.

We use the nonuniform highness convention of [4, Definition 2]: $\mathbf{d}$ is *high for Scott rank* α if every pair of isomorphic computable structures of rank α has a $\mathbf{d}$-computable isomorphism. The program may depend on the pair. Our convention for Scott rank is likewise that of [4, Definition 1]: $\mathrm{SR}(\mathcal{M})$ is the least α such that every finite tuple orbit is definable, without parameters, by a $\Sigma^{\mathrm{in}}_\alpha$ formula. In particular, $\mathrm{SR}(\mathcal{M}) \leq \kappa$ if every finite tuple orbit has a computable infinitary definition of some

computable complexity. Neither the defining formulas nor their complexities need be uniform in the tuples.

All computable structures have domain ω, after a computable recoding when necessary. A computable tree is a prefix-closed decidable subset of $\omega^{<\omega}$, and $[T]$ denotes its set of infinite paths. Uniformity in a tree means uniformity in an index for its characteristic function.

Theorem 1. *Uniformly from a computable tree $T \subseteq \omega^{<\omega}$, one can construct computable structures $\mathcal{A}_T, \mathcal{B}_T$ in a fixed finite language such that:*

1. *Every finite tuple orbit in either structure has a computable infinitary definition. In particular, $\mathrm{SR}(\mathcal{A}_T), \mathrm{SR}(\mathcal{B}_T) \leq \kappa$.*
2. *$\mathcal{A}_T \cong \mathcal{B}_T$ if and only if $[T] \neq \varnothing$.*
3. *A path in T computes an isomorphism $\mathcal{A}_T \to \mathcal{B}_T$, and every such isomorphism computes a path in T. Both conversions are uniform in an index for T.*

The new step is the normal form in Lemma 2. We write a path in binary and attach numerical bounds by which each entry must have been decoded. For a fixed binary prefix, increasing these bounds preserves extendibility. Dickson's lemma then gives a finite basis for the extendible bound vectors. Thus extendibility is computable on each fixed level, although the finite bases need not be obtainable uniformly. This yields the local ordinal bounds required for the Scott analysis.

We combine this recoding with the classical Boolean group-tree construction of Makkai and Morozov, as presented in [5, §3 and Theorem 3.1]. Finite sets of nodes form Boolean groups, and parity over parent fibres defines the projections between levels. Naming different root vectors in two copies forces an isomorphism to encode a nonzero coherent translation, from which a path can be recovered. We give a fixed finite-language presentation and explicit orbit definitions, so that the rank bound and the uniform reductions can be checked directly. Sections 2–4 prove Theorem 1; Section 5 derives the highness equivalence.

2 A levelwise normal form for tree path problems

For a tree S, write S_n for its n-th level and put

$$\mathrm{Ext}(S)_n = \{\sigma \in S_n : (\exists X \in [S])\, \sigma \preceq X\}.$$

Here $\preceq$ denotes the initial-segment relation. A node is *extendible* if it lies on an infinite path, and *dead* otherwise. Write S_σ for the subtree of extensions of σ, with root σ. On a well-founded subtree we use

$$\mathrm{rk}_S(\sigma) = \sup\{\mathrm{rk}_S(\tau) + 1 : \tau \text{ is a child of } \sigma\},$$

so leaves have rank zero. Extendible nodes are assigned rank ∞, greater than every ordinal. We use the same convention for other rooted trees and forests.

Lemma 2. *Uniformly from a computable tree T, one can construct a computable tree $S \subseteq \omega^{<\omega}$ and uniform computable conversions between paths in T and paths in S such that, for each fixed n, the set $\mathrm{Ext}(S)_n$ is computable. The last assertion is not uniform in n.*

Proof. Code $f \in \omega^\omega$ by the binary real

$$X_f = 0^{f(0)}1\,0^{f(1)}1\,0^{f(2)}1\cdots.$$

For a finite binary string η, let $R_T(\eta, i)$ mean that η contains at least $i+1$ occurrences of 1, and that the numbers of zeros in its first $i+1$ completed blocks form a node of T. This predicate is computable and remains true when η is extended.

Consider

$$Q_T = \{(X, b) \in 2^\omega \times \omega^\omega : (\forall i)\, R_T(X \upharpoonright b(i), i)\}.$$

Every member of Q_T computes a path in T: the conditions ensure infinitely many delimiters, and every decoded initial segment belongs to T. Conversely, from $f \in [T]$ compute X_f and

$$b_f(i) = \sum_{j \le i} (f(j) + 1).$$

Then $(X_f, b_f) \in Q_T$.

To represent Q_T by a computable tree, regard a node of length n as a pair $(\rho, u) \in 2^n \times \omega^n$, coding its j-th coordinate by $2u(j) + \rho(j)$. Put (ρ, u) in S exactly when

$$(\forall i < n)\big[u(i) \le n \implies R_T(\rho \upharpoonright u(i), i)\big]. \tag{1}$$

Membership is decidable. If a prefix has length $m < n$, every check it requires has $i < m$ and $u(i) \le m$; that same check was required at length n, with the same binary prefix of length $u(i)$. Thus S is prefix-closed. Its root is the empty sequence, even when T is empty.

Under the coordinate coding, $[S] = Q_T$. Indeed, for a branch $(X, b) \in [S]$ and a fixed i, choose $n \ge \max\{i+1, b(i)\}$. Equation (1) then checks $R_T(X \upharpoonright b(i), i)$. Conversely, every initial segment of a member of Q_T passes all the required checks. This proves the claimed uniform conversions between path sets.

Fix n and $\rho \in 2^n$, and define

$$U_\rho = \{u \in \omega^n : (\rho, u) \in \mathrm{Ext}(S)_n\}.$$

The set U_ρ is upward closed in the coordinatewise order. If $(X, b) \in Q_T$ extends (ρ, u) and $v \ge u$, replace the first n coordinates of b by v, leaving the rest unchanged. The predicates R_T remain true because increasing a decoding bound only extends the binary string being inspected. The resulting member of Q_T extends (ρ, v).

By Dickson's lemma [3], every upward closed subset of ω^n has finitely many minimal elements. We recall the argument. Every infinite sequence of natural numbers has an infinite nondecreasing subsequence: either some value occurs infinitely often, or the tails are unbounded and one can choose a strictly increasing subsequence. Applying this successively in the finitely many coordinates shows that every infinite sequence in ω^n contains two terms, in their original order, that are coordinatewise comparable. Hence antichains, and in particular sets of minimal elements, are finite. Moreover, every element of an upward closed set lies above a minimal element, since it has only finitely many predecessors in ω^n.

Let M_ρ be the finite set of minimal elements of U_ρ, allowing the empty set. Then

$$u \in U_\rho \quad \iff \quad (\exists v \in M_\rho)(\forall j < n)\, v(j) \le u(j). \tag{2}$$

For this fixed n, there are only 2^n choices of ρ. Their finite bases can therefore be included as constants in a decision procedure for $\mathrm{Ext}(S)_n$. No procedure for finding these bases as n varies is needed or asserted. □

Lemma 3. *For the tree S in Lemma 2 and each n, there is a computable ordinal β_n such that*

$$\mathrm{rk}_S(\sigma) < \beta_n \qquad \textit{for every } \sigma \in S_n \setminus \mathrm{Ext}(S)_n.$$

Proof. Fix n. By Lemma 2, the dead nodes on level n form a computable set. Form a rooted tree W_n whose immediate successors are coded by these nodes, with a copy of S_σ rooted at the successor coded by σ. Membership in W_n is computable: first test whether the successor code denotes a dead level-n node, and then use the decision procedure for S. The tree W_n is well-founded, since an infinite path would lie in one of the attached dead subtrees. Its Kleene–Brouwer ordering is a computable well-order. Its order type plus one is a computable ordinal strictly greater than the rank of every attached subtree, and may be used as β_n. The decision procedure for the dead nodes, and hence the resulting bound, is only asserted to exist separately for each n. □

3 The Boolean group construction

We now use the group-tree construction from [5, §3] to turn paths into isomorphisms. The construction itself uses only membership in S; the local information from Lemma 3 will enter the Scott analysis in the next section.

Let ϵ be the empty sequence, so $S_0 = \{\epsilon\}$. For each n, let

$$G_n = \mathcal{P}_{\mathrm{fin}}(S_n)$$

be the Boolean group of finite subsets of S_n, with symmetric difference as addition. We keep the level in the code of each vector; in particular, the zero vectors $0_n = \varnothing$ at different levels are distinct elements. Define

$$\pi_n : G_{n+1} \longrightarrow G_n, \qquad \pi_n(F) = \big\{\sigma \in S_n : |\{\tau \in F : \tau \restriction n = \sigma\}| \text{ is odd}\big\}. \tag{3}$$

Parity shows that π_n is a homomorphism. Put $V = \coprod_n G_n$, and define $p : V \to V$ by $p \restriction G_{n+1} = \pi_n$ and $p \restriction G_0 = \mathrm{id}$.

To obtain a fixed finite language, we name translation relations by indices rather than using a separate symbol for each translation. Adjoin an index sort I, a copy of $(\omega, 0, s)$. For $k \in \omega$, write $\overline{k} = s^k(0)$ for its numeral in this sort. Let $\ell : V \to I$ be the level map, so $\ell(F) = \overline{n}$ for $F \in G_n$. Fix, independently of S, a computable bijection

$$e : \omega \times \mathcal{P}_{\mathrm{fin}}(\omega^{<\omega}) \longrightarrow \omega$$

with computable inverse. Define the ternary relation D by

$$D(\overline{e(n,H)}, F, F') \quad \Longleftrightarrow \quad H \subseteq S_n, \quad F, F' \in G_n, \quad F' = F \mathbin{\triangle} H. \tag{4}$$

It is false on all other triples. In particular, an index that decodes to (n, H) with $H \not\subseteq S_n$ names the empty relation.

The language consists of the unary predicate I, constants $0, c$, unary functions s, p, ℓ, and the ternary relation D. We use a one-sorted presentation on $I \sqcup V$: extend s by the identity on V, and extend p, ℓ by the identity on I. Hereafter $V(x)$ abbreviates $\neg I(x)$. The index sort is rigid, since every one of its elements is the value of a numeral.

These interpretations are computable uniformly from S. A finite set belongs to G_n exactly when all of its members belong to S_n, which is a finite computable test. Thus the tagged domain has a computable coding as an infinite decidable subset of ω; its increasing enumeration gives a computable bijection with ω, uniformly from S. This also makes the codes of the vectors 0_n uniformly computable. No extendibility information is used in this presentation.

Let $\mathcal{A}$ and $\mathcal{B}$ have the same reduct just described, with

$$c^{\mathcal{A}} = 0_0, \qquad c^{\mathcal{B}} = 1_0 := \{\epsilon\}.$$

Thus the two elements of G_0 are 0_0 and 1_0.

Lemma 4. *An isomorphism $\mathcal{A} \to \mathcal{B}$ is exactly a map fixing I pointwise and acting on each G_n as*

$$F \longmapsto F \triangle h_n, \qquad h_n \in G_n, \qquad \pi_n(h_{n+1}) = h_n, \qquad h_0 = 1_0. \tag{5}$$

Automorphisms of either structure have the same description with $h_0 = 0_0$.

Proof. An isomorphism f fixes I pointwise because it fixes 0 and preserves s. Preservation of ℓ then forces it to preserve each level. Set $h_n = f(0_n)$. Applying f to $D(\overline{e(n, F)}, 0_n, F)$ gives $f(F) = h_n \triangle F$. Since $p(0_{n+1}) = 0_n$, preservation of p yields $\pi_n(h_{n+1}) = h_n$. For an isomorphism $\mathcal{A} \to \mathcal{B}$, preservation of c gives $h_0 = 1_0$. For an automorphism of either structure, the translation on G_0 fixes its named element and hence is zero.

Conversely, a coherent sequence (h_n) defines a bijection on each level. The homomorphism identity gives

$$\pi_n(F \triangle h_{n+1}) = \pi_n(F) \triangle h_n,$$

so the resulting map preserves p. Translating both vector arguments in (4) leaves their difference unchanged, so it preserves D. It preserves $I, s, \ell, 0$ by its definition, and the condition at level zero ensures preservation of c. □

Lemma 5. *Paths in S and isomorphisms $\mathcal{A} \to \mathcal{B}$ compute one another uniformly.*

Proof. Given a path $X \in [S]$, put $h_n = \{X \upharpoonright n\}$. These vectors satisfy (5), and the corresponding isomorphism is computable from X.

Conversely, an isomorphism f computes the finite sets $h_n = f(0_n)$. Start with $\sigma_0 = \epsilon \in h_0$. If $\sigma_n \in h_n$, the coherence equation says that an odd, and therefore nonzero, number of members of h_{n+1} have parent σ_n. Choose the one with least code and call it σ_{n+1}. Then $\sigma_n \preceq \sigma_{n+1}$ and $\sigma_n \in S_n$ for all n, so their union is a path through S. The procedure uses only evaluations of f and finite searches. □

4 The Scott rank bound

An orbit condition must determine whether a finite translation can be extended coherently through all higher levels. The point of the normal form is that, at each fixed level, failure of such an extension is witnessed by a tree rank below one computable ordinal.

On the vector sort define the proper child relation

$$C(x, y) \quad \iff \quad V(x) \wedge V(y) \wedge p(y) = x \wedge y \neq x.$$

This makes V a forest with roots $0_0, 1_0$. The subtree rooted at $F \in G_n$ is its *lift tree*: a child of a vector is a vector on the next level projecting to it. Each lift node has a unique ancestor chain back to F.

Lemma 6. *A vector $F \in G_n$ has an infinite branch in its lift tree if and only if $F \subseteq \mathrm{Ext}(S)_n$. If $\sigma \in F \setminus \mathrm{Ext}(S)_n$, the lift tree of F is well-founded and has rank at most $\mathrm{rk}_S(\sigma)$.*

Proof. Suppose every member of F is extendible. Choose a path extending each member, and at each higher level take the finite set of nodes on these paths. Paths extending distinct level-n nodes remain distinct, so the resulting vectors project to one another and give an infinite lift branch. For $F = \varnothing$, use the all-zero branch.

Now fix a dead node $\sigma \in F$. We define a map from the lift tree of F into S_σ that takes children to children. Send its root to σ. If a lift node E has been sent to $\tau \in E$ and E' is a child of E,

the parity equation (3) ensures that E' contains a child of τ. Send E' to the child with least code. Uniqueness of ancestor chains makes this prescription consistent. An infinite lift branch would map to an infinite branch through S_σ, which is impossible. Finally, well-founded induction gives the rank inequality: if q is this map and the inequality holds at the children of a lift node E, then

$$\mathrm{rk}(E) = \sup_{E' \text{ child of } E} \big(\mathrm{rk}(E') + 1\big) \leq \sup_{E' \text{ child of } E} \big(\mathrm{rk}_S(q(E')) + 1\big) \leq \mathrm{rk}_S(q(E)).$$

Apply this at the root. □

For each computable ordinal α, rank at least α in this forest has a computable infinitary definition. Fixing computable ordinal notations, define

$$\begin{aligned} H_0(x) &\equiv V(x), \\ H_{\alpha+1}(x) &\equiv \exists y \, (C(x,y) \wedge H_\alpha(y)), \\ H_\lambda(x) &\equiv V(x) \wedge \bigwedge_j H_{\lambda_j}(x) \qquad (\lambda \text{ a nonzero limit}), \end{aligned} \tag{6}$$

where $(\lambda_j)_j$ is the computable cofinal sequence supplied by the chosen notation for λ. The recursion along the notation makes the limit conjunctions effective and yields a computable infinitary formula of computable complexity. Induction on α shows that H_α holds exactly at nodes of rank at least α, with infinite rank included. At a successor step, rank at least $\alpha+1$ is equivalent to having a child of rank at least α; at a limit, it is equivalent to having rank at least every λ_j.

Choose β_n as in Lemma 3 and set $\theta_n(x) = H_{\beta_n}(x)$. By Lemma 6, for $F \in G_n$ we have

$$\theta_n(F) \quad \Longleftrightarrow \quad F \subseteq \mathrm{Ext}(S)_n. \tag{7}$$

Vectors with extendible support have infinite lifts. Any other vector contains a dead node $\sigma \in S_n$, so its lift rank is at most $\mathrm{rk}_S(\sigma) < \beta_n$. A notation for β_n is chosen only for this particular level; a computable sequence of such notations is not required.

Consider a nonempty tuple of vector-sort elements $a_i = (n_i, F_i)$, and let $N = \max_i n_i$. In $\mathcal{A}$, define

$$\begin{aligned} \varphi_{\bar a}(\bar x) \equiv \exists z \, \big[& V(z) \wedge \ell(z) = \overline{N} \wedge \theta_N(z) \wedge p^N(z) = c \\ & \wedge \bigwedge_i D\big(\overline{e(n_i, F_i)}, p^{N-n_i}(z), x_i\big)\big]. \end{aligned} \tag{8}$$

Here p^k is the k-fold iterate of p, with p^0 the identity. All numerals in the formula are finite terms in $0, s$; the entries of $\bar a$ are not used as parameters.

We verify that (8) defines the orbit of $\bar a$. If an automorphism sends $\bar a$ to $\bar b$, let (h_n) be its coherent translation sequence. Taking $z = h_N$ works: the higher coordinates form an infinite lift branch, so $\theta_N(h_N)$ holds, and $p^N(h_N) = h_0 = 0_0 = c^{\mathcal{A}}$. The translation relations express precisely that $b_i = F_i \triangle h_{n_i}$.

Conversely, suppose $\bar b$ satisfies (8), witnessed by z. Equation (7) and Lemma 6 extend z upward to a coherent sequence. Below level N, use its iterated projections. The condition $p^N(z) = c^{\mathcal{A}} = 0_0$ ensures that the resulting translation is an automorphism, by Lemma 4. The last conjunction in (8) makes it send each a_i to b_i.

Index-sort elements have singleton orbits defined by their numerals. For a mixed tuple, conjoin these singleton definitions with (8) for the vector-sort coordinates. A tuple entirely in I needs only the singleton definitions, and the empty tuple has the tautological orbit definition. Each resulting formula is computable infinitary of computable complexity. Therefore $\mathrm{SR}(\mathcal{A}) \leq \kappa$.

The automorphisms of $\mathcal{B}$ are the same zero-root translations. In $\mathcal{B}$, the condition

$$D(\overline{e(0,\{\epsilon\})},c,w)$$

defines $w = 0_0$, since $c^{\mathcal{B}} = 1_0$. Replacing $p^N(z) = c$ in (8) by

$$D(\overline{e(0,\{\epsilon\})},c,p^N(z))$$

therefore gives the same orbit verification in $\mathcal{B}$, without changing the conclusion about computable infinitary definability. Thus $\mathrm{SR}(\mathcal{B}) \leq \kappa$ as well. Together with Lemmas 2 and 5, this completes the proof of Theorem 1.

5 Highness at the boundary ranks

Corollary 7. *A Turing degree is high for Scott rank ω_1^{CK} if and only if it is high for isomorphism. These conditions are also equivalent to highness for Scott rank $\omega_1^{\mathrm{CK}} + 1$. In particular, the degree requested in Question 1 of [4] does not exist.*

Proof. Let $\mathbf{d}$ be high for Scott rank κ. We first use the padding observation following Definition 2 of [4]. Given isomorphic computable structures of rank at most κ, take their tagged disjoint unions with a fixed computable structure of exact rank κ, whose existence follows from [5]. The tagged sums have exact rank κ, and an isomorphism between them restricts to an isomorphism between the original structures. Hence $\mathbf{d}$ computes isomorphisms for pairs of rank at most κ as well.

For any ill-founded computable tree T, Theorem 1 supplies isomorphic computable structures $\mathcal{A}_T, \mathcal{B}_T$ of rank at most κ. A $\mathbf{d}$-computable isomorphism between them computes a path in T. Thus $\mathbf{d}$ computes a path through every ill-founded computable tree.

For completeness, the passage from this path property to highness for isomorphism is the following implication from [1, Proposition 2.6]. Let $\mathcal{C}, \mathcal{D}$ be isomorphic computable structures in a common computable language, and enumerate the atomic formulas of that language. Form a tree of finite sequences of nested finite injections from $\mathcal{C}$ to $\mathcal{D}$. At stage j, require that the injection cover the first j elements in both domain and range. Require also that the first j atomic formulas agree in truth value on every assignment from its domain and the corresponding image assignment. These are finite computable tests, so the tree is computable and prefix-closed. An isomorphism supplies a path by taking suitable finite restrictions. Conversely, the union along any path is a total bijection, and every atomic formula on every tuple is eventually checked. The union is therefore an isomorphism. Applying the path property gives a $\mathbf{d}$-computable isomorphism $\mathcal{C} \to \mathcal{D}$. The reverse implication, from highness for isomorphism to highness for rank κ, follows directly from the definitions.

Finally, every computable structure has Scott rank at most $\kappa + 1$. Together with the same tagged-sum observation at $\kappa + 1$, this identifies highness for that rank with highness for isomorphism, as noted in [4, §4]. □

The finite bases and ordinal bounds are used only to prove the rank bound. They are not inputs to the construction of the structures or to the extraction of a path. Consequently, for each input tree, the isomorphism program supplied by nonuniform highness can be composed with the uniform extraction without any additional oracle information.

Remark (Exact rank). If T has a path but no hyperarithmetical path, both structures in Theorem 1 have rank exactly κ. They are isomorphic, and if their common rank were computable, the computable-rank upper bound of [4, Lemma 2], together with the padding observation, would give

a hyperarithmetical isomorphism. The extracted path in T would then be hyperarithmetical, a contradiction.

In this case the well-founded node ranks of S have no common computable bound. Indeed, if a computable ordinal γ bounded all of them, the rank-at-least-$\gamma + 1$ test, constructed as in (6) for the child relation of S, would decide extendibility uniformly in the node using a hyperarithmetical oracle. Starting at the root and repeatedly choosing the least extendible child would yield a hyperarithmetical path in S, and hence in T. In particular, the levelwise bounds β_n cannot be bounded by a computable ordinal.

References

[1] Wesley Calvert, Johanna N. Y. Franklin, and Dan Turetsky, *Structural highness notions*, Journal of Symbolic Logic **88**(4) (2023), 1692–1724. doi:10.1017/jsl.2022.35.

[2] Wesley Calvert, Julia F. Knight, and Jessica Millar, *Computable trees of Scott rank ω_1^{CK}, and computable approximation*, Journal of Symbolic Logic **71**(1) (2006), 283–298. doi:10.2178/jsl/1140641175.

[3] Leonard E. Dickson, *Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors*, American Journal of Mathematics **35**(4) (1913), 413–422. doi:10.2307/2370405.

[4] Johanna N. Y. Franklin, Dino Rossegger, and Dan Turetsky, *Structural vs. computational complexity*, arXiv preprint, 2026. arXiv:2606.15196v1.

[5] Julia F. Knight and Jessica Millar, *Computable structures of rank ω_1^{CK}*, Journal of Mathematical Logic **10**(1–2) (2010), 31–43. doi:10.1142/S0219061310000912. Author version: arXiv:math/0508507.

[6] Michael Makkai, *An example concerning Scott heights*, Journal of Symbolic Logic **46**(2) (1981), 301–318. doi:10.2307/2273623.