ON EFFECTIVENESS OF OPERATIONS ON COUNTABLE ORDINALS

JOEL DAVID HAMKINS AND ZHENHAO LI

Abstract. We study the possibility of representing natural operations on countable ordinal numbers by effective procedures that work uniformly in well ordered relations on the natural numbers representing the ordinals, and show there are natural operations that can not be represented in this effective way. The notion that an countable ordinal is easier than another can be formalized as existences of effective procedures that produce representation of the former uniformly from those of the latter, and we give two ways to do so the first of which relates to the well known Medvedev reducibilities on mass problems. Despite expectations to the contrary, we answer affirmatively the question whether there is a procedure to compute a representation of ω_1^{CK} uniformly from any representation of $\omega_1^{CK} + \omega$.

1. Introduction

Ordinal numbers has been playing important roles in mathematical logic for more than one century since Cantor introduced them in the last decade of 19th century. A standard modern set theoretical definition of ordinal numbers can be found in [Jec03]. Cantor's original definition can be found in [Can97].

Representations of ordinal numbers have studied from the beginning. Cantor's normal form is a most well known representation system for ordinals; see [Jec03]. From the 30's to 60's of the 20th century, researchers including Church, Kleene, Markwald, Rogers and Spector studied effective representation systems of ordinals. Ordinals that have computable representation are well known and have been called constructive ordinals and recursive ordinals, and they all have ordinal notations in Kleene's $\mathcal{O}$; see [Kle38], [Mos38], and [Rog67]. A lot of research has also been done on the interplay of computability and linear orderings. Results can be found in the survey [Dow98].

The research of the first author has been partially supported by grants from the National Science Foundation, the Simons Foundation and the CUNY Research Foundation. This research was partially done whilst the authors were visiting fellows at the Isaac Newton Institute for the Mathematical Sciences in the programme 'Semantics and Syntax'. The second author is thankful for the discussions on this topic with Benedikt Löwe, Russell Miller, Antonio Montolbán, and Philip Welch. Benedikt Löwe also provided many helpful concrete suggestions on the writing of this paper.

Ordinals that have computable representation form an initial segment of countable ordinals which is known as ω_1^{CK}. However, in ZermeloFraenkel set theory (ZF), we know that all countable ordinals can be represented by well ordered relations on the natural numbers, or ω. Details about these representations of ordinals are provided in Section 2 and 3. In Section 3 we provide a direct proof that arithmetical sets do not represent more ordinals than computable ones, which is a consequence of Spector's theorem that hyper-arithmetical sets do not represent more ordinals than computable ones [Spe55].

Nevertheless little is known about the effectiveness of natural operations on countable ordinals in terms of existence of computable procedures that work uniformly in the representations of ordinals. We formally define an effectiveness notion in Section 4 and show that some naturally defined ordinal functions are effective while some are not. For example, we show that the three basic operations of ordinal arithmetic are effective and the max function is not. Using games, we introduce a mechanism to discover operations on ordinals that are not effective.

We also explore the notion that one ordinal is easier to represent than another ordinal. In Section 5 we require that ordinal α is easier than β only if there is a procedure that computes a representation of α uniformly from any representation of β. It turns out this definition coincide with the well known Medvedev reducibility if we see countable ordinals as mass problems. To understand this degrees difficulties of representing ordinals, the second author asked the following challenge question last year, and it received attention from a number of set theorists and computability theorists.

Question 1. *Can one compute a representation of ω_1^{CK} uniformly from any representation of $\omega_1^{CK} + \omega$?*

Many of those who considered the question, including ourselves and a group of researchers at the conference "The Incomputable" held at Chicheley Hall in June 2012, conjectured that the answer would be negative. In this paper, however, we prove in Theorem 37 that in fact there is such a procedure.

In Section 6 we provide another way to compare the difficulties of representing ordinals. We are able provide affirmative answers to some questions whether two ordinals have the same difficulties whereas we don't know the answers to these questions if we use the way we compare them in Section 5.

2. PRELIMINARIES

2.1. **Relative computability and Turing functionals.** Relative computation is the main tool for our work and we use oracle Turing machines as the models of it. For definition of Turing machines and oracle Turing machines, see [Soa87] and [Soaar]. Although a standard definition of oracle Turing machine uses only one oracle tape, it is convenient to work with Turing machines with multiple oracle tapes. (It is easy to check that for

each $\alpha \leq \omega$, a Turing machine with α oracle tapes on α many oracles can be simulated by a Turing machine with one oracle tape on one oracles. If we assume some standard computable coding of α many reals into one real, an index for the one-tape machine can be found effectively in an index for the α-tape machine, and an index for an α-type Turing machine that codes α many oracles into one can be found effectively.)

For each $\alpha \leq \omega$, we have a family of Turing machines having α many oracle tape and an effective indexing for this family. For any sequence $X_0, X_1, ...$ of reals of length α If $\Phi^{X_0,X_1,\cdots}$ is total, i.e., $\Phi^{X_0,X_1,\cdots}(i)\downarrow$ for every i, we use $\Phi(X_0, X_1, ...)$ to denote the unique $Y \in 2^\omega$ such that $Y(i) = \Phi^{X_0,X_1,\cdots}(i)$ for all i, and we say Y is computable by Φ from $X_0, X_1, ...$ So $\Phi(X_0, X_1, ...)$ can be seen as the infinite output (which is a real) when $X_0, X_1, ...$ are given as inputs when the oracle Turing machine Φ is seen as a function which takes reals as inputs, and produce reals as outputs, which is called an α-ary *Turing functional*. We will often call Turing machines or Turing functionals *procedures* for convenience. For a real A and a class $\mathcal{C}$ of reals, we say A is computable uniformly from $\mathcal{C}$ if there is a Turing functional Φ such that $\Phi(B) = A$ for any $B \in \mathcal{C}$.

For a finite string $\sigma \in 2^{<\omega}$, $\Phi^\sigma(n)\downarrow$ if and only the computation of Φ on input n using σ as the oracle halts and Φ does not read beyond σ during the computation, and $\Phi^\sigma(n) = y$ if and only if $\Phi^\sigma(n)\downarrow$ and y is the output.

2.2. **Coding binary relations on ω.** It is well known that binary relations on ω can be coded to subsets of ω, namely the reals. The most natural way to code binary relations on ω into reals(subsets of ω) is to use pairing functions. A pairing function $f : \omega \times \omega \to \omega$ is increasing if it satisfies that $f(a, b) < f(c, d)$ if $a = c$ and $b < d$, or $a < c$ and $b \leq d$. Most pairing functions commonly used are increasing and moreover computable, such as the Cantor pairing function. We fix a computable increasing pairing function $\langle\ \rangle : \omega \times \omega \to \omega$.[1] After fixing our pairing function, we have a unique way to code binary relations into reals. We say a binary relation R is coded by A if $(a, b) \in R \leftrightarrow \langle a, b\rangle \in A$.

We define the *field* of A or field(A) to be the set of all x's such that $\langle x, b\rangle \in A$ or $\langle b, x\rangle \in A$ for some b. Elements of field(A) are called *points* of A. For any two points of A a and b, we say a is below b in A if $\langle a, b\rangle \in A$. We also write $a <_A b$ for $\langle a, b\rangle \in A$. If a is in the field of A we use $A^{<a}$ to denote the order A below point a, namely $\{\langle x, y\rangle \in A : x <_A a \wedge y <_A a\}$. In a similar way we define $A^{\geq a}$ and so on.

Proposition 2. *There is a total Turing functional that computes the field of any real code uniformly from it.*

Proof. To decide if $x \in$ field(A), just check if $\langle x, x\rangle \in A$. □

[1]It can be easily checked that results in this paper do not depend on which computable pairing function to use. So we don't give any specific one here.

By the Truth table theorem of Nerode, this tells us that field(A) is truth table reducible to (or truth table computable from) A.

Proposition 3. *None of the set of the least point, the set of successor points, or the set of limit points, of any real code, can be uniformly computed from the real code.*

Proof. Suppose we can compute the singleton set of the least point of A which is $\{a\}$ from A. The computation that determines a as the least point will use only a finite initial segment of the oracle A. But we can easily add a new lower point below a (to code a different ordinal) which does not change that finite initial segment of A. The same procedure will still determine that a is the least point in the new code since the part of oracle used in the computation is not changed. Contradiction.

Similarly, if an algorithm says that b is a successor (of a in A), then we can choose an infinite sequence of (large enough) numbers and insert them between a and b to code a different ordinal on which the same procedure also decides that b is a successor, contradiction.

□

2.3. **Sequences and trees.** We fix some standard coding $\omega^{<\omega}$ into ω, for example using prime factorization. We use $\langle a_0, a_1, ..., a_n\rangle$ to denote the code of the sequence of natural numbers $a_0a_1...a_n$, and use σ_e to denote the sequence coded by e. Given a string σ of length n, we use $\sigma(i)$ to donate the i-th number in σ for $i < n$. To distinguish the Cantor space from the Baire space, we call in this paper members of 2^ω (binary) strings and members of ω^ω suquences (of numbers). It is easy to see that the initial segment relation $\preceq$ is computable on codes of sequences, i.e., there is a procedure ϕ such that $\phi(x, y) = 1$ if the both x and y are codes of sequences and the sequence coded by x is an initial sequence of that coded by y and $\phi(x, y) = 0$ otherwise. A tree is a set of strings or sequences that is closed under taking initial segments. For each tree T, $[T]$ is the set of infinite paths in T. Note that each tree T can be coded into a set of natural numbers or equivalently a member of 2^ω via the coding of sequences. We say a tree is computable if its code is computable.

3. Real codes of countable ordinals

Countable ordinals can be represented by non-strict well orderings on subsets of ω. If A codes a well ordering and has order-type α we say A is a real code of α. For each countable ordinal α, let $\mathrm{WO}_{=\alpha}$ be the set of all real codes of α, and $\mathrm{WO}_{\leq\alpha}(\mathrm{WO}_{\geq\alpha})$ be the set of all real codes of ordinals no larger(smaller) than α. Note that for each infinite α, $|\mathrm{WO}_{=\alpha}| = \mathfrak{c}$, the cardinality of the continuum. The set of all real codes of countable ordinals is WO. Let Wel be the index set $\{e : \phi_e$ is total and is the characteristic function of some set in WO$\}$. It is easy to see that WO is a Π^1_1 class and Wel is a Π^1_1 set. Moreover Wel is Π^1_1-complete. See [Rog67, p.222, 397].

A countable ordinal α is *computable* if and only if there is a computable A such that $A \in \mathrm{WO}_{=\alpha}$. It has been known that computable ordinals form a proper initial segment of ω_1 and the order type of this initial segment is the Church-Kleene ordinal ω_1^{CK} which is also the least ordinal that does not have an ordinal notation (ordinals having ordinal notations are called constructive ordinal, so computable ordinals and constructive ordinals are the same; see [Rog67, p.205–222]). The fact that Wel is Π^1_1 gives us that $\mathrm{WO}_{\omega_1^{\mathrm{CK}}}$ and $\mathrm{WO}_{\geq\omega_1^{\mathrm{CK}}}$ are also Π^1_1 classes.

It seems natural to define computably enumerable ordinals, or arithmetical ordinals. But it turns out that going up in the arithmetical hierarchy does not give more power to express ordinals; by the results of Spector in[Spe55] ordinals having hyper-arithmetical real codes are the same as ordinals having computable real codes. We mention that Theorem 54 in Section 7 provides among other conclusions a more elimentary and direct proof that arithmetical reals do not code more ordinals than computable reals.

We are mostly interested in those properties of ordinals that can be computed from all their real codes uniformly. Before moving to the next section, we present here some results regarding computing the least point, successor points and limit points from real codes.

In Proposition 3 we have seen that the least point is not computable uniformly from the code. But there could be a procedure Φ that on A outputs A' such that A' and A code the same ordinal, and the least point of A' is computable uniformly from A' by some procedure Ψ. Our next theorem shows it is the case.

Proposition 4. *There is procedure that uniformly computes from any real code A a real code B such that it codes the same ordinal as A does and the least point of B is* 0.

Proof. The idea is to copy all points of A except the least one without using 0 as any point and put 0 below all points in the copy. While copying A from left to right above, we use the current least point a as a filter and only copy points above a until we find a new least point a' in A that is below a, and then we use a' as the filter and only copy points above it and so on. Whenever we need to represent a point a of A in the copy, we always choose a new number p that is larger than all that have been used and all information about a will be coded into information about p in further copying.

□

Proposition 5. *For each natural number n, there is a procedure that uniformly computes from any real code A a real code A' such that it codes the same ordinal as A does and the i-th least point of A' is $i-1$ for each $i \leq n$.*

Proof. Use the same idea as in Proposition 4. □

But we do not have similar results for successors and limits.

Proposition 6. *There are no procedures Φ and Ψ such that Φ uniformly computes from any real code A a real code A' such that it codes the same ordinal and Ψ computes the set of limit points of A' from A'.*

Proof. Suppose Φ and Ψ exist. Let A code $\omega + 1$. Let a be the unique point in $\Psi \circ \Phi(A)$. Let n to be the length of the initial segment of A used in the computation that determines $a \in \Psi \circ \Phi(A)$. Now let B be a code of ω such that its first n bits are identical to those of A. Then $a \in \Psi \circ \Phi(B)$ but $\Psi \circ \Phi(B)$ should be empty. □

Proposition 7. *There are no procedures Φ and Ψ such that Φ uniformly computes from any real code A a real code A' such that it codes the same ordinal and Ψ computes the set of successors of A' from A'.*

Proof. If this is true, then Proposition 6 would have been true as well since the set of limit points of A' is computable uniformly from the set of successor points of A' and the field of A'. □

4. Uniformly computable ordinal functions

Let Φ be an oracle Turing machine using n oracle tapes, or, a n-ary Turing functional. If $\Phi(X_0, ..., X_{n-1})$ is defined and is a real code of α_n whenever X_i is a real code of α_i for $i < n$, we say Φ *uniformly computes* α_n from $\alpha_0, \alpha_1, ..., \alpha_{n-1}$.

Definition 8. *Let Φ be a n-ary Turing functional and f be a partial n-ary ordinal function, we say Φ uniformly computes f if Φ uniformly computes $f(\alpha_0, ..., \alpha_{n-1})$ from $\alpha_0, ..., \alpha_{n-1}$ for any $\alpha_0, ..., \alpha_{n-1}$ that are in the domain of f; we say Φ strongly uniformly computes f if Φ uniformly computes β from $\alpha_0, ..., \alpha_{n-1}$ if and only $\alpha_0, ..., \alpha_{n-1}$ are in the domain of f and $\beta = f(\alpha_0, ..., \alpha_{n-1})$.*

We say f is (strongly) uniformly computable if it is (strongly) uniformly computed by some functional Φ. A computable ordinal function is maximal *if it cannot be extended to another one.*

Obviously each Turing functional strongly computes some ordinal function which might be the empty function. We are interested in the question what functions on ordinals, or, ordinal functions that occur in ordinary mathematics are uniformly computable and which are not. We intuitively think that "natural" ordinal functions should be uniformly computable. The next theorem shows that the three basic operations of ordinal arithmetic are computable. But it is not always the case. We will see later that there are simple ordinal functions that are defined naturally but not uniformly computable.

Theorem 9. *The operations addition, multiplication and exponentiation of ordinal arithmetic are all uniformly computable.*

Proof. Addition. Given a real code A of α and a real code B of β, copy A using even numbers and copy B using odd numbers, and by putting the copy of B above the copy of A we get a real code of $\alpha + \beta$.

Multiplication. Given a real code A of α and a real code B of β, to get a real code of $\alpha \cdot \beta$ put $\langle a, b\rangle$ below $\langle a', b'\rangle$ if b is below b' in B, or $b = b'$ and a is below a' in A.

Exponentiation. A finite support function from β to α is a function from β to α such that only finitely many elements of β is mapped to nonzero values. It is well known that α^β is the order type of the finite support functions from β to α under the reverse lexical order. Suppose A is a real code of α and B is a real code of β. We want to construct the set of functions from β to α with finite support and code the reverse lexical order of this set of functions into C. We fix a computable coding of finite sequence of natural numbers into natural numbers. Let A' be the output of the procedure in Proposition 4 when applied to A. A finite sequence of pairs $\langle\langle x_0, y_0\rangle ... \langle x_n, y_n\rangle\rangle$ is in the field of C if $x_i \in \text{field}(A')$ and $x_i \neq 0$ for each $i \leq n$ and y_i is below y_j in B for each $i < j \leq n$. We put $\langle\langle a_0, b_0\rangle ... \langle a_m, b_m\rangle\rangle$ below $\langle\langle x_0, y_0\rangle ... \langle x_n, y_n\rangle\rangle$ in C if, after removing common pairs in both sequences, the left sequence becomes empty, or the last pair in the left sequence is $\langle a', b'\rangle$, the last pair in the right sequence is $\langle x', y'\rangle$ and b' is below y' in B, or, $b' = y'$ and a' is below x' in A. It is easy to check C is a real code of α^β. □

Theorem 10 (folklore)**.** *For each n, the function $\alpha + n \mapsto \alpha$ is uniformly confirmable. The result is uniform in n.*

Proof. While copying A from left to right above, we use the current top n points that have been read as a filter and only copy points below these points. Whenever we need to represent a point a of A in the copy, we always choose a new number p that is larger than all that have been used so far and all information about a in A will be coded into information about p in the copy. □

We will show in the rest of this section that many naturally defined ordinal functions are not uniformly computable. We provide a general method to show an ordinal function is not uniformly computable. This method involve a notion of P-distinguishable pairs of ordinals where P is some property. The notion is introduced by a special game which is described below.

Given a property P and a pair of ordinals (α, β), there is a game (P, α, β) that is played alternately between two players where the fist player makes the first move, and that lasts ω steps. Each player starts with empty string and extends it during the game. The second player wins the game if and only if his final infinite string is a real code of α or β, and it must be α when the final string of the first player is infinite and does not satisfy P and it must be β when the final string of the first player is infinite and satisfies P. If the second player has a winning strategy in this game, it means that he can say to the first player "I will extend my string into a real code of either

α and β, and if you can extend your string into a real that does not satisfy P then mine will codes α, and if you can extend your string into a real that satisfies P then mine will codes β."

First we introduce the concept of a pseudo functional.

Definition 11. *Given a property P and a pair of ordinals (α, β), we say (α, β) is effectively P-distinguishable if the second player in the game (P, α, β) has a computable winning strategy.*

We give several results about effectively P-distinguishable pairs for certain P's.

Lemma 12. *Let P be the property of "having at least one point", i.e., let $P(X)$ be the Σ^0_1 statement that $\exists n[n \in \text{field}(X)]$. Then $(\omega+1, \omega)$ is effectively P-distinguishable.*

Proof. Fix a computable code B of $\omega + 1$, and a computable code C of ω. The following is a computable winning strategy for the second player in the game $(P, \omega + 1, \omega)$. Continue copying B as long as the string played by the first player so far does not contain any point, and once the first player's string contains a point, start to copy C above all the points copied from B. □

Lemma 13. *Let P be the property of "having at least two points", i.e., let $P(X)$ be the Σ^0_1 statement that $(\exists n, m \in \text{field}(X))[m \neq n]$. Then $(\omega, \omega + \omega)$ is effectively P-distinguishable.*

Proof. Similar to the proof of Lemma 12. □

Lemma 14. *Let P be the property of "having infinitely many points", i.e., let $P(X)$ be the Π^0_2 statement that $\forall n \exists m[m > n \wedge m \in \text{field}(X)]$. Then $(\omega + \omega, \omega)$ is effectively P-distinguishable.*

Proof. Fix a computable code B of $\omega + \omega$. The following is a computable winning strategy for the second player in the game $(P, \omega + \omega, \omega)$. Continue copying B as long as the first player does not add any new point to his string, and once the first player adds a new point, start a new copying of B above all the points in the current string. It easy to see that if the first player only adds finitely many points then the second player's final string will be a real code of $\omega + \omega$; if first player only adds infinitely many points then the second player's final string will be a real code of ω.

□

Lemma 15. *Let P be the property of "being empty or having a maximal point", i.e., let $P(X)$ be the Σ^0_2 statement that $\forall x[x \notin \text{field}(X)] \vee \exists n(\forall m \neq n)[\langle n, m\rangle \notin X]$. Then $(\omega, \omega + \omega)$ is effectively P-distinguishable.*

Proof. Fix a computable code B of $\omega + \omega$. The following is a computable winning strategy for the second player in the game $(P, \omega + \omega, \omega)$. Continue copying B as long as the first player does not add any new point above

all existing points in his string, and once the first player adds a new point above all existing points in his string, start a new copying of B above all the points in the current string. It easy to see that if the first player only adds a maximal point finitely often then the second player's final string will be a real code of $\omega + \omega$; if first player only adds a maximal point infinitely often then the second player's final string will be a real code of ω.

□

Let P be a property on the reals. We say α *satisfies* P if every real code of it satisfies P; and say α *avoids* P every real code of it does not satisfy P.

Theorem 16. *Suppose a pair (α, β) is effectively P-distinguishable, and λ avoids P and γ satisfies P. Then there is no Turing functional that uniformly compute γ from α and β from λ.*

Proof. Suppose there is a procedure Φ that computes γ from α and β from λ. By definition of P-distinguishable pair, we know that the second player has computable winning strategy in the game (P, α, β).

First we show that there is a computable Y such that Y is the finial string of the second player in the game (P, α, β) where he follows the computable strategy and the first player always play $\Phi(\sigma)$ where σ is the string of the second player at that time. It is easy to see that Y is computable and the final string of the first player is infinite by the assumption about Φ and in fact is $\Phi(Y)$ which is also computable.

Now $\Phi(Y)$ is either does not or does satisfy P. Suppose that the former case is true. Then Y must be a code of α by the condition of the theorem and hence $\Phi(Y)$ is a real code of γ. But since γ satisfies P, we have reached a contradiction. Similar argument works for the latter case. □

Corollary 17. *There is no Turing functional that uniformly computes ω from $\omega + \omega$ and n from ω.*

Proof. Let P such as in Lemma 14. Apply Theorem 16.

□

Corollary 18. *The function $\alpha \mapsto$ the least λ such that $\lambda + \omega = \alpha$ is not uniformly computable.*

Corollary 19. *There is no Turing functional that uniformly computes* 1 *from $\omega + 1$ and* 0 *from ω.*

Proof. Let P be such as in Lemma 12. Apply Theorem 16. □

Corollary 20. *The function that maps $\lambda + n$ to n where n is finite is not uniformly computable.*

Corollary 21. *There is no Turing functional that uniformly computes* 1 *from ω and* 2 *from $\omega \cdot 2$.*

Proof. Let P be such as in Lemma 13. Apply Theorem 16. □

Corollary 22. *The function $\omega \cdot \beta \mapsto \beta$ is not uniformly computable.*

The above results show that Cantor's normal form of ordinals are not uniformly computable, even if we only consider a very small initial segment of all countable ordinals.

Theorem 23. *The function $\{(\omega, \omega+1), (\omega+\omega, \omega+\omega)\}$ is not uniformly confirmable.*

Proof. Let P be such as in Lemma 15. Apply Theorem 16. □

Corollary 24. *The function* $\max(\alpha, \beta)$ *is not uniformly computable.*

Proof. By Theorem 23, the unary function

$$g(\alpha) = \begin{cases} \omega+1 & \text{if } \alpha \leq \omega+1 \\ \alpha & \text{otherwise} \end{cases}$$

is not uniformly computable. Hence f cannot be uniformly computable. □

Corollary 25. *For each $n > 0$, The supremum function*

$$(\alpha_0, ..., \alpha_n) \mapsto \sup\{\alpha_0, ..., \alpha_n\}$$

is not uniformly computable.

Proof. If this is computable, then the binary max will also be computable, contradiction. □

Corollary 26. *The function* $\sup_{i<\omega} \alpha_i$ *is not uniformly confirmable.*

Now the question is whether the function $\sup_{i<\omega} \alpha_i$ uniformly confirmable for sequences $\{\alpha_i\}_{i<\omega}$ that does not contain a maximal element. We show some results that partially answer this question.

Definition 27. *Let Φ be a Turing functional with infinitely many inputs and λ be a limit ordinal. We say Φ uniformly limit-computes λ if given codes A_i's of α_i's such that α_i's are infinite ordinals and unbounded in λ, Φ outputs a code of λ. A limit ordinal λ is uniformly limit-computable if it is uniformly limit-computed by some functional Φ.*

It follows directly that the function $\sup_{i<\omega} \alpha_i$ is uniformly confirmable when restricted to sequences $\{\alpha_i\}_{i<\omega}$ such that $\sup_{i<\omega} \alpha_i$ is limit-computable and $\{\alpha_i\}_{i<\omega}$ is unbounded in $\sup_{i<\omega} \alpha_i$.

Theorem 28. *All additively closed limits can be uniformly limit-computed by one single Turing functional.*

Proof. Suppose λ is additively closed, A_i codes α_1 for each i, and $\{\alpha_i\}_{i<\omega}$ is unbounded in λ. Effectively fix ω many pairwise disjoint infinite sets $I_{i<\omega}$(for example let p_n is the n-th prime number and I_i be the set $\{(p_i)^n : n > 0\}$), and for each n Φ copy the n-th input A_n using points from I_n. Meanwhile Φ makes sure that all points from I_n are above all points from I_m for all $m < n$. The output codes an ordinal no greater than λ because λ is additively closed and no less than λ because $\{\alpha_i\}_{i<\omega}$ is unbounded in λ. □

Corollary 29. *There is a Π^1_1-complete real code of ω_1^{CK}.*

Proof. We already know Wel the set of indices of computable real codes is Π^1_1-complete.

Let Φ be as in the proof of Theorem 28. There is a procedure Ψ such that $A = \Psi(\text{Wel}) = \Phi(A_1, A_2, ...)$ where A_i is the empty set if $i \notin \text{Wel}$ and is the code of $\alpha + 1$ and having 0 as its least point where α is the ordinal having a code computable by φ_i if $i \in \text{Wel}$. Note that we can get A_i effectively from i by Proposition 4. Since ω_1^{CK} is additively closed, and real codes computed by elements of Wel are unbounded in ω_1^{CK}, A is a real code of ω_1^{CK}. It is also true that Wel is computable from A because it is computable from field(A). If we want to know whether $i \in \text{Wel}$, we only need to ask if the 0 in the i-th copy of ω (namely I_i in the proof of Theorem 28) is a point in A. □

When the limit is not additively-closed, what can we say about the sup function? Can the sup be uniformly computed when the input are an increasing sequence of ordinals? While we do not know the answer, we shall see below that the answer is yes in a implicit way.

Lemma 30. *There is a Turing functional Φ such that if A and B code the same ordinal then $\Phi(A, B)$ is a tree on pairs of natural numbers whose unique infinite path contains the isomorphism from A to B, and $\Phi(A, B)$ has no infinite path otherwise.*

Proof. Let i^A be the ith element in the field of A, and let Φ be the following procedure. Given a sequence of pairs $\sigma = \langle\langle x_0, y_0\rangle ... \langle x_{n-1}, y_{n-1}\rangle\rangle$, it is in $\Phi(A, B)$ if and only if $x_i \in \text{field}(A)$ and $y_i \in \text{field}(B)$ for each $i < n$, $x_{2i} = i^A$ for each $i < n/2$ and $y_{2i+1} = i^B$ for each $i < (n-1)/2$, and σ is a partial isomorphism between A and B.

□

Given a countable set or a set S of ordinals, if any subset of S has order-type ω, then we say S is ω-like. It is easy to see that if S is ω-like then it is unbounded and $\sup(S) = \sup(S')$ for any infinite subset S'.

Theorem 31. *There is a functional Ψ such that given codes A_i's of an ω-like sequence of infinite ordinals α_i's whose limit is some λ, Ψ outputs a tree T whose infinite paths exist and are all real codes of λ.*

Proof. First we construct a tree G of guesses, and the guesses are being verified during the construction so that in the end the only infinite path in the tree. The guesses are about an ω-long increasing chain of some of the α_i's, and for each adjacent $\alpha_k < \alpha_j$ in the chain, an isomorphism from A_k to a initial segment of A_j below a point a_j in A_j. The details are as follows.

Let Φ be the same as in Lemma 30. First we decide sequences of length 1 in G. These are $\langle n, 0, \langle\rangle\rangle$ for all natural number n. Suppose δ of length n is already in G, we add δb to G if and only if

(1) σ_b is of length $n+3$ and $\sigma_b(1)$ is in the field of $A_{\sigma_b(0)}$ (remember that σ_b is the sequence coded by b);
(2) for each $1 < i < n+1$, $\sigma_{\sigma_b(i)}$ is a proper extension of $\sigma_{\sigma_{\delta(n-1)}(i)}$ in the tree of $\Phi(A_{\sigma_{\delta b(i-2)}(0)}, A^{<\sigma_{\delta b(i-1)}(1)}_{\sigma_{\delta b(i-1)}(0)})$ (that is, the i-th number in the sequence coded by b codes a sequence that is a proper extension of that coded by the i-th number in the sequence coded by the last number of δ, in the tree of $\Phi(A_x, A_y^{<a})$ where x is the first number in the sequence coded by the $(i-2)$-th number in δb, y is the first number in the sequence coded by the $i-1$-th number in δb, and a is the second number in the sequence coded by the $(i-1)$-th number in δb.)

Let Φ' be the Turing functional that computes G from the inputs. Now it is easy to see how to compute a real code of λ from any infinite path X of G. Construct a code that in effect puts $A_{\sigma_{X(0)}(0)}$ at the bottom, adds on top $A^{\geq\sigma_{X(1)}(1)}_{\sigma_{X(1)}(0)}$, then adds on top $A^{\geq\sigma_{X(2)}(1)}_{\sigma_{X(2)}(0)}$ and then $A^{\geq\sigma_{X(3)}(1)}_{\sigma_{X(3)}(0)}$ and so on. Let Θ be such a functional.

Now construct a computable tree T in Cantor space. The idea is that T wants to produce infinite paths that are isomorphic copies of real codes in $\Theta[\Phi'(A_0, A_1, ...)]$ while it guesses about the relevant computations and these guesses and the verifications of them are coded into the points in the branches of T. Infinite paths in T will contain correct guesses about all computation evolved. Formally we put δ into T if

(1) each element of field(σ) is $\langle z, j\rangle$ for some z such that $\sigma_z \in \Phi'(A_0, A_1, ...)$ and some j that is in field$(\Theta(\sigma_z))$;
(2) for each j, there is at most one z such that $\langle z, j\rangle$ is in the field of δ. If $j < k$ and both $\langle z, j\rangle$ and $\langle z', k\rangle$ are in the field of δ, then $\sigma_z \prec \sigma_{z'}$;
(3) the function taking $\langle j, z\rangle$ to j is a finite isomorphism from δ to a finite order that is coded by some initial segment of $\Phi(\tau)$ for some $\tau \in \Phi'(A_0, A_1, ...)$ such that $\sigma_z \preceq \tau$ for all z that appear in δ.

Note that all the conditions can be checked effectively. For the last condition, the search for τ is bounded and hence computable. Now T is computable in the given codes A_i's and all members of $[T]$ are isomorphism to $\Theta(X)$ where X is in $[\Phi'(A_0, A_1, ...)]$, and let Ψ be the procedure that constructs T.

□

5. The Medvedev degrees of ordinals

Medvedev defined in [Med55] the degrees of difficulty of mass problems. A *mess problem* is a set of infinite sequences of natural numbers, namely a subset of the Baire space ω^ω. In this paper, we only deal with mass problems on Cantor space. The motivation is that each subset of the space can be seen as a collection of solution to some problem. In our setting, $\mathrm{WO}_{=\alpha}$ is a mass

problem because it is the set of solutions to the problem of "coding ordinal α". Mass problem $\mathcal{A}$ is Medvedev reducible to mass problem $\mathcal{B}$, or $\mathcal{A} \leq_M \mathcal{B}$, if there is a Turing functional Φ such that $\Phi(X) \in \mathcal{A}$ for any $X \in \mathcal{B}$. The equivalent classes of the resulting equivalence relation $\equiv_M$ are called *degrees of difficulty*, and ordered by $\leq_M$ they form a distributive lattice known as the Medvedev lattice. For basics and background see [Med55], [Rog67], and [Sor96].

We say β is Medvedev reducible to α, or $\beta \leq_M \alpha$, if $\mathrm{WO}_{=\beta}$ is Medvedev reducible to $\mathrm{WO}_{=\alpha}$. It is easy to see that $\beta \leq_M \alpha$ if and only if the ordinal function $\{(\alpha, \beta)\}$ is uniformly confirmable. Intuitively $\beta \leq_M \alpha$ means that it is not harder to code β than to code α since we can compute a code of β from any code of α using a single procedure. We are interested in $\omega_1 / \equiv_M$. We call members of $\omega_1 / \equiv_M$ *Medvedev degrees* of (countable) ordinals.

It can be easily observed that ordinals below ω_1^{CK} constitute one Medvedev degree, namely the Medvedev degree of the empty set.

Proposition 32. *The set of computable ordinals form the bottom Medvedev degree, that is formally $\omega_1^{CK} = \mathbf{0}_M$.*

Proof. Since each $\alpha \in \omega_1^{CK}$ has a computable real code $R \in \mathrm{WO}_{=\alpha}$, there is a trivial oracle Turing machine that output R on any oracle (as input) and hence $\alpha \leq_M \beta$ for any countable β. □

It is easy to see $\alpha + \beta \leq_M \alpha$ for any $\beta < \omega_1^{CK}$. The same holds for $\times$ and exponentiation in place of $+$. (See Theorem 9.)

Theorem 33. *For each ordinal α and each n, $\alpha \equiv_M \alpha + n$.*

Proof. By Theorem 10. □

Corollary 34. *The Medvedev degrees of ordinals are closed under adding finite ordinals.*

Now it is natural to ask the following question:

Question 35. *Are the Medvedev degrees of ordinals closed under adding ω?*

The first test question is whether $\omega_1^{CK} \leq_M \omega_1^{CK} + \omega$. The idea used to prove the last proposition is not applicable to this question. Surprisingly we still get an affirmative answer. But the proof uses a very simple algebraic property, namely the additive closure property, of ω_1^{CK}. An ordinal α is *additive closed* if $\beta + \gamma < \alpha$ for each $\beta < \alpha$ and each $\gamma < \alpha$. Note that an additive closed ordinal must be a limit.

Lemma 36. *There is a procedure Φ such that if λ is uniformly limit-computable, A_n is a real code of α_n for each n and $\sup\{\alpha_n : n \in \omega\} = \lambda$, then $\Phi(A_0, A_1, ...)$ is a real code of λ.*

Proof. For each $k = \langle n, m \rangle$, construct C_k that codes the order-type of the predecessors of m in the code A_n of α_n. It is easy to see that the C_n's are

unbounded in λ. Let Ψ be the functional that uniformly limit-computes λ. The output of Φ is $\Psi(C_0, C_1, ...)$ which is a code of λ.

□

Theorem 37. *There is a computable functional Φ that uniformly confirms $(\lambda + \omega, \lambda)$ if λ is uniformly limit-computable.*

Proof. Suppose we are given a code A for $\lambda+\omega$ where λ is a limit ordinal and are to construct a code C for λ. Note that the points that are below a limit ordinal in the code A has order-type λ precisely, since $\lambda + \omega$ has a largest limit ordinal. Note also that the limit points of the code A are precisely those points that infinitely often receive a new immediate predecessor in the enumeration of the order as it is revealed by A.

The idea is to compute for each n a code A_n of α_n. The computation will make sure that each α_n is at most λ, and exactly one of them will be equal to λ. Then we can use the procedure provided by Lemma 36 to get a real code of λ.

Now we only need to show that the A_n's works as we promised. The idea is to guess n is a limit points in A and construct A_n so that it codes α_n, the order-type of the predecessors of n in A. It is easy to see each α is at most λ and for the n that is the λ-th point in A, it will be exactly λ. If n turns out not to be a limit, then α_n will be finite.

Construction of A_n: Start copying A in the following way. Every time n gains a new immediate predecessor in A, this is evidence that the guess about n is correct, and so we allow all the points below n in A that have been revealed so far to be copied. Only allowed points of A can be copied. This happens infinitely often just in case n really does code a limit ordinal, and we will have allowed the entire cut below n into the code of α_n, which will be the order-type of that initial segment. Otherwise, when n has an immediate predecessor, this will eventually be revealed and so we will copy only finitely many points from A, so the resulting code will be that of a finite ordinal.

□

Corollary 38. *For each n, there is a computable functional Φ uniformly confirms $(\lambda + \omega \cdot n, \lambda)$ if λ is uniformly limit-computable.*

Proof. Modify the last proof. Fix a computable enumeration of ω^n. Let the strategy for i guess that all numbers in the i-th n-tuple of numbers are limit points in the code A. □

Corollary 39. *For each finite n and each finite k, there is a computable functional Φ uniformly confirms $(\lambda + \omega \cdot n + k, \lambda)$ if λ is uniformly limit-computable.*

Proof. By Corollary 38 and Theorem 10. □

Corollary 40. *For each finite n and finite k, $\omega_1^{CK} \equiv_M \omega_1^{CK} + \omega \cdot n + k$.*

Question 41. *Is there a natural characterization of the Medvedev degrees of ordinals?*

The structure of the Medvedev degrees of ordinals is not very clear so far. The following theorem shows that it is not trivial.

Theorem 42. *Assume that ω_1 is regular. Then $\leq_M$ does not preserve the natural order on ordinals.*

Proof. Suppose it does. Then $\omega_1^{CK} \leq_M \alpha$ for every α greater than ω_1^{CK}. Since there are ω_1 many such α's, there is a (least) Turing functional Φ_1 that uniformly confirms an ordinal function f_1 the range of which is $\{\omega_1^{CK}\}$ and the domain of which has size ω_1 (we can safely assume the domain is maximal). Take θ_1 to be the least ordinal in the domain of f_1. Then we get a (least) Turing functional Φ_2 that uniformly confirms an ordinal function f_2 the range of which is $\{\theta_1\}$ and the domain of which has size ω_1 and is a subset of the domain of f_1 (again we can safely assume the domain of f_2 is maximal with these properties).

The goal is to construct an ω_1-chains of Turing functionals $\{\Phi_\alpha\}_{\alpha<\omega_1}$ and ordinal functions $\{f_\alpha\}_{\alpha<\omega_1}$ such that

(1) the domain of each f has size ω_1 and $\mathrm{dom}(f_\alpha) \supseteq \mathrm{dom}(f_\beta)$ for $\alpha < \beta$;
(2) the range of each f is a singleton and $\mathrm{ran}(f_\alpha) \neq \mathrm{ran}(f_\beta)$ for $\alpha \neq \beta$.
(3) Φ_α uniformly confirms f_α.

At successor steps we repeat the above construction. At countable limit step λ we first take the interception of the domains of the ordinal functions constructed so far (which form a chain under $\supseteq$) and call it T. Then T must have size ω_1 and let θ_λ be the least element of T, so we can get a (least) Turing functional Φ_λ that uniformly confirms an ordinal function f_λ the range of which is $\{\theta_\lambda\}$ and the domain of which has size ω_1. This can be repeated for ω_1 times since ω_1 is regular. Hence the desired ω_1-chains of Turing functionals $\{\Phi_\alpha\}_{\alpha<\omega_1}$ and ordinal functions $\{f_\alpha\}_{\alpha<\omega_1}$ are constructed.

Notice that from the three conditions above we conclude that $\Phi_\alpha \neq \Phi_\beta$ for $\alpha \neq \beta$. But this is a contradiction because there are only countably many Turing functionals.

□

Question 43. *Are there two incomparable ordinals with respect to $\leq_M$?*

6. The uniformly implicit reducibility

We introduce another reducibility relation on ordinals. An ordinal α is *uniformly implicit reducible* to an ordinal β, or $\alpha \leq_{u.i.} \beta$, if there is a Turing functional Φ such that for any real code X of β, $\Phi(X)$ is a tree whose infinite paths are all real codes of α. Note that $\leq_M$ implies $\leq_{u.i.}$.

It is easy to check that $\leq_{u.i.}$ is reflexive. We show that $\leq_{u.i.}$ is transitive on the mass problems of coding ordinals.

Theorem 44. *If* $\mathrm{WO}_{=\alpha} \leq_{u.i.} \mathrm{WO}_{=\beta}$ *and* $\mathrm{WO}_{=\beta} \leq_{u.i.} \mathrm{WO}_{=\gamma}$, *then* $\mathrm{WO}_{=\alpha} \leq_{u.i.} \mathrm{WO}_{=\gamma}$.

Proof. We use the same technique as in the proof of Corollary 31. We use p_n to denote the n-th prime number and use σ_z to denote the finite string coded by z.

Suppose $\mathrm{WO}_{=\alpha} \leq_{u.i.} \mathrm{WO}_{=\beta}$ via Φ and $\mathrm{WO}_{=\beta} \leq_{u.i.} \mathrm{WO}_{=\gamma}$ via Ψ. We effectively construct tree T with arbitrary $X \in \mathrm{WO}_{=\gamma}$ that witness $\mathrm{WO}_{=\alpha} \leq_{u.i.} \mathrm{WO}_{=\gamma}$. The idea is that T wants to produce infinite paths that are isomorphic copies of $[\Phi\Psi(X)]$ while it guesses about the computations and these guesses and the verifications of them are coded into the points in the branches of T. Infinite paths in T will contain correct guesses about all computation evolved. Formally, we put σ into T if the following are satisfied.

(1) Each element of field(σ) is p_z^j for some z such that $\sigma_z \in \Psi(X)$ and some j that is in field$(\Phi(\sigma_z))$
(2) For each j, there is at most one z such that p_z^j is in the field of σ. If $j < k$ and both p_z^j and $p_{z'}^k$ are in the field of σ, then $\sigma_z \prec \sigma_{z'}$.
(3) The function taking p_z^j to j is an order isomorphism from σ to some initial segment of $\Phi(\tau)$ for some $\tau \in \Psi(X)$ such that $\sigma_z \preceq \tau$ for all z that appear in σ. Note that all the conditions can be checked effectively. For the last condition, the search for τ is bounded and hence computable.

It is easy to see that T is computable uniformly from the given code X and all members of $[T]$ are isomorphism to some member of $[\Psi(X)]$. □

The equivalent classes of the resulting equivalence relation $\equiv_{u.i.}$ on are called *u.i. degrees of ordinals*, and ordered by $\leq_{u.i.}$.

We give an example of u.i. reducibility.

Proposition 45. *For any* λ *that has unbounded limit points,* λ *is u.i. reducible to* $\lambda + \omega \cdot \omega$, *in particular* $\omega_1^{CK} \leq_{u.i.} \omega_1^{CK} + \omega \cdot \omega$.

Proof. We call a point a in a real code C *local maximal* if a is C-larger than any point that has occurred before a, i.e., $\forall b{\in}\mathrm{field}(C)[b < a \to b <_C a]$. It is easy to see that a any real code C, C has infinitely many local maximal points if and only if C codes a limit ordinal. Let lm be an oracle Turing machine such that $\mathrm{lm}^C(i)$ is the i-th local maximal point in C. So for real code C of an infinite limit ordinal, $\mathrm{lm}(C)$ is the infinite sequence of local maximal points in C. Note that points in $\mathrm{lm}(C)$ are unbounded in C.

The important observation is that λ is the largest point α in θ such that there are unbounded many limit points below it, and above it is a computable ordinal $\omega \cdot \omega$. We fix a computable code W of $\omega \cdot \omega$.

Reserve two copies of ω for each point a in field(C). We will compute a tree T_a such that at the end T_a will have infinite paths if an only if a has unbounded limit points below it. At the same time we compute a tree F_a such that at the end F_a have infinite paths if and only if $C^{\geq a}$ codes $\omega \cdot \omega$.

The idea is that T_a is a tree of guesses of limit points below a and F_a is a tree of partial isomorphisms from W to $C^{\geq a}$, and the verification of the guesses are also coded into the trees. Wrong guesses will eventually stop receiving extensions and so infinite paths in the trees only contain the correct guesses and the verifications.

At each leave i, a node p corresponds the guess that p is a limit points C-above $\mathrm{lm}^{C^{<a}}(i)$. Our construction will satisfy that if the sequence $\{b_i\}_{i<n}$ in the tree T_a then $b_i >_C \mathrm{lm}^{C^{<a}}(i)$ for each i. The construction makes sure that an infinite branch $\{p_i\}_{i\in\omega}$ is in T_a if and only each p_i receives infinite many new immediate predecessors during reading C from left to right.

Suppose that a branch $\{b_i\}_{i<n}$ is already computed from C with use k, i.e., only the first k bits of C has been read when $\{b_i\}_{i<n}$ is added in the T_a. Then $\{b_i\}_{i\leq n}$ is added in T_a with use l if

(1) for each $i < n$, b_i has received a new immediate predecessor between the k-th bit and the l-th bit of C;
(2) $\mathrm{lm}^{C^{<a}}(n)$ occurs in $C\upharpoonright l$;
(3) b_n occurs in $C\upharpoonright l$ and $b_n >_C \mathrm{lm}^{C^{<a}}(n)$.

The tree F_a is constructed the same way as in Lemma 30. Given a sequence of pairs $\sigma = \langle\langle x_0, y_0\rangle ... \langle x_{n-1}, y_{n-1}\rangle\rangle$, it is in F_a if and only if $x_i \in \mathrm{field}(W)$ and $y_i \in \mathrm{field}(C^{\geq a})$ for each $i < n$, $x_{2i} = i^W$ for each $i < n/2$ and $y_{2i+1} = i^{C^{\geq a}}$ for each $i < (n-1)/2$, and σ is an isomorphism between the x's ordered by W and the y's ordered by C.

It is easy to verify that T_a and F_a satisfies all requirements.

Now we effectively construct a tree S that witness the u.i. reducibility. We put σ in S if the following are true.

(1) For some $a \in \mathrm{field}(C)$, each element of $\mathrm{field}(\sigma)$ is $\langle a, u, z, j\rangle$ for some u and z such that $\sigma_u \in F_a$ and $\sigma_z \in T_a$.
(2) The j's that occur in σ is an initial segment of natural numbers and for each j_i there is unique u_i and z_i such that $\langle a, u_i, z_i, j_i\rangle$ occur in σ.
(3) For $j_i < j_{i'}$, it is the case that $\sigma_{u_i} \prec \sigma_{u_{i'}}$ and $\sigma_{z_i} \prec \sigma_{z_{i'}}$.
(4) The map that takes $\langle a, u, z, j\rangle$ to $j^{C^{<a}}$ is a partial order isomorphism from σ to $C^{<a}$.

□

We can generalise the above proof into the next theorem. We call a property P on ordinals *implicitly verifiable* if there is a procedure Φ such that for any α, if α satisfies P then $\Phi(X)$ has infinite paths for any $X \in \mathrm{WO}_\alpha$ while if α does not satisfy P then $\Phi(X)$ has no infinite path for any $X \in \mathrm{WO}_\alpha$.

Theorem 46. *Let α be a computable ordinal. If λ satisfies a implicitly verifiable property P and $\lambda + \beta$ does not satisfy P for any $\beta < \alpha$, then $\lambda \leq_{u.i.} \lambda + \alpha$.*

Proof. Easy by the definition of implicitly verifiable properties and the proof of Proposition 45. □

Theorem 47. *For any ordinal λ and any computable ordinal α, $\lambda \leq_{u.i.} \lambda \cdot \alpha$.*

Proof. We fix a computable code W of α and reserve a special symbol ∞.

We construct a tree S of sequences of pairs of natural numbers. We put σ in S if the following are satisfied.

(1) The first pair in σ is $\langle \infty, b_0 \rangle$ for some $b_0 \in \text{field}(W)$ and for $i > 0$ the i-th pair in σ is $\langle i^W, b_i \rangle$ for some b_i in field(W) or $b_i = \infty$.
(2) The first $|\sigma|$ bits of W satisfies that there is no point between $\langle i^W, b_i \rangle$ (when b_i is ∞ it means there is no point above i^W, with the first pair it means there is no point below b_0).

It is not hard to see that S has a unique infinite path which codes the least point and the successor relation in W.

Suppose C is an arbitrary code of $\lambda \cdot \alpha$. For each two pairs $\langle x_0, y_0 \rangle$ and $\langle x_1, y_1 \rangle$ only consisting points in C, we construct a tree $T^C_{(x_0,y_0),(x_1,y_1)}$ such that is has infinite path if and only if $C^{\geq x_0, <y_0}$and $C^{\geq x_1, <y_1}$ codes the same ordinal. This can be done by Lemma 30. Now we construct a tree T by putting σ in it when the following are satisfied.

(1) The σ is a finite sequence of finite sequences of codes of finite strings such that the i-th sequence in σ is of length $i + 2$.
(2) The first component in each sequence in σ is a z such that σ_z is a sequence of pairs such that the $j + 1$-th pair is $\langle j^W, a_{j+1} \rangle$ for some a_{j+1} in C.
(3) The second component in each sequence in σ is a u such that σ_u is a sequence of length $i + 1$ in the tree S, and the 0-th pair of σ_z is $\langle b_0, d_0 \rangle$ for some d_0 in C where b_0 is such that $\langle \infty, b_0 \rangle$ is the first pair in σ_u.
(4) For each $k < |\sigma| + 2$ the $k + 2$-th component is in $T^C_{z \cdot u(k+1), z \cdot u(k+2)}$ where $z \cdot u(n)$ is the pair $\langle a, b \rangle$ such that $\langle x, a \rangle$ and $\langle y, b \rangle$ occur in σ_z and $\langle x, y \rangle$ is the $(n + 1)$-th pair in σ_u.
(5) For any $i < j$, it is the case that for each $k < i + 2$, the k-th component in the i-th sequence of σ codes a string that is a proper initial segment of that coded by the k-th component of the j-th sequence of σ.
(6) The point d_0 is the bottom point in $C\restriction|\sigma|$.

The tree T is a tree of guesses about the points that represent $\lambda \cdot \gamma$ for $\gamma < \alpha$, and the verification of the guesses are also coded into T itself. Wrong guesses will eventually stop receiving extensions and so infinite paths in T only contain the correct guesses. It is not hard to verify that T has infinite paths if and only if C codes an ordinal of the form $\lambda \cdot \alpha$ for some λ.

Now we effectively construct a tree T' that witness the u.i. reducibility. Intuitively we get T' by reprocessing T. We get a branch of T' by extracting a partial code of λ from each branch of T while all information in that

branch of T are coded into the field of that partial code of T'. So at the end all infinite path in T' comes from infinite paths in T, and since infinite paths in T contain correct informations, the corresponding infinite paths in T' will be correct code of λ. Formally, we put σ in T' if the following are true.

(1) Each element of field(σ) is p^j_z for some z such that $\sigma_z \in T$.
(2) The j's that occur in σ form an initial segment of natural numbers and for each j_i there is unique z_i such that $p^{j_i}_{z_i}$ occur in σ. For $j_i < j_{i'}$, it is the case that $\sigma_{z_i} \prec \sigma_{z_{i'}}$.
(3) The map that takes p^j_z to the $j^{C^{\geq a,<b}}$ is a partial order isomorphism from σ to $C^{\geq a,<b}$ where $(a, b) = v{\cdot}u(1)$ where v is the first component and u is the second component in the first sequence of σ_z.

□

Corollary 48. *For any computable ordinal α, the property of "having the form $\beta \cdot \alpha$" is implicitly verifiable.*

Proof. By the proof of the theorem. □

A natural question to ask is whether $\leq_{u.i}$ preserve the natural order of ordinals. The answer is negative. We can prove it by an easy modification of the proof of Theorem 42.

Theorem 49. *Assume that ω_1 is regular. Then $\leq_{u.i.}$ does not preserve the natural order on ordinals.*

Question 50. *Is $\leq_{u.i.}$ strictly weaker than $\leq_M$?*

7. DEGREES OF CLASSICAL REDUCIBILITIES IN REAL CODES OF ORDINALS

In this section we focus on studying the Turing degrees, bounded Turing degrees, truth table degrees, and hyper-degrees of real codes of ordinals.

Let $\leq$ be any reducibility relation that is not stronger than the truth table reducibility $\leq_{tt}$. It is easy to show that for infinite countable ordinal α, $\mathrm{WO}_{=\alpha}$ contains arbitrarily high degrees under $\leq$, since for any given infinite set S there is a real code A of any ordinal such that field$(A) = S$ and the functional field is truth table computable as shown in Proposition 2. So there is no upper bound of degrees in $\mathrm{WO}_{=\alpha}$ under $\leq$.

Given a set S, we call the set of all reals X such that $S \leq X$ the cone above S. It is easy to see that for each $\alpha \leq \beta$ and $B \in \mathrm{WO}_{=\beta}$ there is a $A \in \mathrm{WO}_{=\alpha}$ such that $A \leq_{tt} B$ (Let A be $B_{<a}$ where a is the point in B such that $B_{<a}$ has order type α). Hence under $\leq$ that is not stronger than the truth table reducibility $\leq_{tt}$, WO_α lies in a cone if and only if $\mathrm{WO}_{\geq\alpha}$ lies in it. If all the real codes of α is in the cone above S, then the degree of S is a lower bound for the degrees of real codes of α. For $\leq$ not stronger than the Turing reducibility $\leq_T$, if we know such a lower bound is a real code α, we can completely characterise the degrees of real codes of α.

Theorem 51. *Let $\leq$ be any reducibility relation that is not stronger than the Turing reducibility $\leq_T$. If all the real codes of α are in the cone above a real code A of α, then the degrees (under $\leq$) of real codes of α are exactly the degrees of the reals in the cone above A.*

Proof. Fix a S such that $A \leq S$. It is easy to effectively construct a A' such that the map that takes the n-th element in field(A') to the n-th element field(A) is an isomorphism from A' to A, and field$(A') = S$. So A' and S has the same degree. □

A nice consequence of this theorem is a complete characterisation of the hyper-degrees of real codes of incomputable ordinals. For more background on hyper-dgrees and hyper-arithmatical reducibility, see [Rog67, s.16.6]

Corollary 52. *The hyper-degrees of real codes of ω_1^{CK} are exactly those hyper-degree that are above the non-trivial hyper-degree of Π^1_1-complete sets.*

Proof. By Corollary 29, there is a Π^1_1-complete real code of ω_1^{CK} and we only need to show that $\mathrm{WO}_{\geq\omega_1^{CK}}$ is contained in the hyper-cone above a Π^1_1-complete set.

Recall that Wel is the index set $\{e : \phi_e$ is total and is the characteristic function of a real code of some ordinal $\}$ and is Π^1_1-complete. It suffices to show that Wel is in $\Delta^1_1(X)$ for any $X \in \mathrm{WO}_{\geq\omega_1^{CK}}$. Since Wel is Π^1_1, it is in $\Pi^1_1(X)$ for any X. We only need to check that Wel is in $\Sigma^1_1(X)$ for $X \in \mathrm{WO}_{\geq\omega_1^{CK}}$. Note that $x \in$ Wel if and only if $\exists F$ [F codes an isomorphism from ϕ_x to $X_{<a}$ for some point a in X]. The sentence in the parentheses is certainly arithmatical in X. □

Question 53. *Is it true that the hyper-degrees of the real codes of each ordinal form a hyper-cone?*

For Turing reducibility $\leq_T$, we know that the real codes of incomputable ordinals cannot be contained in any non-trivial Turing cone, since Richter showed in [Ric81] that for each incomputable linear order there are two real codes of it such that the Turing degrees of them form a minimal pair. Now it is natural to ask: if the degrees of real codes of incomputable ordinals contain minimal Turing degrees? The answer is yes. In fact each ordinal has continuum many real codes of minimal Turing degrees. First we get the following useful theorem.

Theorem 54. *For each set X and each ordinal α such that $\alpha = \omega{\cdot}\alpha$, if α has a real code A that is computable from X' then it also has a real code B that is computable from X. In fact there is an effective procedure that computes an X-index of B from an X'-index of A, uniformly in all X's, A's, and α's.*

Proof. Without loss of generosity, take X to be the empty set 0. Let $A = \Phi_e(0') \in \mathrm{WO}_{=\alpha}$. We will construct a computable B that codes α. From the

construction it will be clear that the index for B can be found effectively in e.

We know by the Limit Lemma of Shoenfield that

$$(\forall x{\in}\omega)\ A(x) = \lim_{n\to\infty} f(x,n)$$

for some computable function f and such a f can be found effectively in e; see [Soaar] and[Sho59].

Let O_n be the longest initial segment of $A_n = \{x < n : f(x,n) = 1\}$ that codes a linear ordering. We construct B by finite extension. We use B_s to denote the initial segment of B constructed before stage $s+1$. During the construction we will maintain a table T of pairs of numbers and use T_s to denote the table before stage $s+1$. If a pair (a,b) is in the table at stage s, it means informally that stage s hopes that the point a in B represent the point b in A.

To each stage s of the construction of B we associate a number n_s. After each stage s, T_s will see an order preserving injection from O_{n_s} to B_s. Let $s_0 = 0$. In stage $s+1$, find the least m such that O_m is different from O_{n_s} and let $n_{s+1} = m$. Let O' be the common initial segment of O_{n_s} and $O_{n_{s+1}}$. For each points a, b in B_s such that T_s witnesses that a and b represent two successive points in O', let $B_{s,>a,<b}$ be the set of all points between x and y in B_s. Let $B_{s,<}(B_{s,>})$ be the set of all points in B_s that are below the point which T_s sees to represent the least(largest) point in O'. Do the following:

(1) Let B' be B_s and T' be the table of pairs in T_s such that the represented points are in O'.
(2) (a) If $O_{n_{s+1}}$ has i more points below all the points in O' than what B_s has in $B_{s,<}$, extend B' in such a way that it has i new points blow the points in B_s that represent points in O' but above all the points already in $B_{s,<}$.
    (b) If $O_{n_{s+1}}$ has i more points above all the points in O' than what B_s has in $B_{s,>}$, extend B' in such a way that it has i new points above all points in B_s.
    (c) For each two points a and b such that they represents two successive points in O' according to T_s, if $O_{n_{s+1}}$ has i more points between the two represented points in O' than what B_s has in $B_{s,>a,<b}$, extend B' in such a way that it has i new points blow the point represented by b and above all the points in $B_{s,>a,<b}$.
(3) Extend T' so that it contains an order preserving injection from $O_{n_{s+1}}$ to B'. This is possible by step (2). Let $T_{s+1} = T'$ and $B_{s+1} = B'$.

To check that B codes the same ordinal as A, first notice that for each point in A its representative in B eventually stabilises because for any n, $f{\restriction}n$ will not change after finitely many stages. It may be the case that at the end we have points in B that are not representing any points of A. By construction, these points can have order type at most ω below the point

that represents the least points in A, or between two points that represent two successive points in A. But since $\alpha = \omega{\cdot}\alpha$, B still codes the same ordinal as A does.

□

Remark 55. *In* [AK00]*, they show that if A is a Δ^0_3 linear ordering then $\omega \cdot A$ has a computable copy. The proof involves the n-systems that is not easy to understand, and it is not uniform and hence does not imply Theorem 54. We believe Theorem 54 also holds for linear orderings. The only thing to be verified is that if $A = \omega \cdot A$ then the ordering obtained by replacing each point in A with any order type $\leq \omega$ will be isomorphic to A.*

An immediate application of Theorem 54 is to get the following fact that follows from the result of Spector in [Spe55] that incomputable ordinals cannot have hyper-arithmetical real codes.

Corollary 56 (Spector 1955)**.** *For $\alpha \geq \omega_1^{CK}$, α does not have arithmetical real codes.*

Proof. Clearly $\omega_1^{CK} = \omega{\cdot}\omega_1^{CK}$. It is easy to see that if we have a arithmetical real code for some ordinal $\beta \geq \omega_1^{CK}$, we also have a arithmetical real code for ω_1^{CK}. Then we have a code for ω_1^{CK} that is computable from $0^{(}n)$ for some finite n. So by applying Theorem 54 for n times we get a computable real code for ω_1^{CK} which is a contradiction. □

Another thing we need is the Cooper jump inversion theorem proved in [Coo73].

Theorem 57 (Cooper 1975)**.** *For each Turing degree $\mathbf{a} \geq \mathbf{0}'$, there is a minimal Turing degree $\mathbf{m}$ such that $\mathbf{m}' = \mathbf{a}$.*

Now we are ready to prove that there are continuum many real codes for each incomputable ordinal.

Theorem 58. *Each countable ordinal has continuum many real code having minimal Turing degrees.*

Proof. First note that for each computable ordinal α, and each set S there is a real code of α having the same Turing degree as S. That is because α has a computable real code A and we can effectively construct a real code A' such that A' is isomorphic to A and field$(A') = S$. It follows from Cooper jump inversion theorem that there are continuum many minimal degrees. Hence each computable ordinal has continuum many real code of minimal Turing degrees.

Assume that α is incomputable and $\alpha = \omega \cdot \alpha$. Then α has continuum many real code that is Turing above $0'$. For each such real code B, Cooper jump inversion theorem tells us that there is a set X having a minimal Turing degree such that X' computes B. But by Theorem 54, we know X computes a B' such that B' codes the same ordinal. Since X has a minimal Turing degree, B' has the same minimal Turing degree.

Finally if α is incomputable and $\alpha < \omega \cdot \alpha$, then $\alpha = \beta + \gamma$ where $\beta = \omega \cdot \beta$ and $\gamma < \beta$ (use Cantor's normal form). It is easy to see that if for each real code of β having a minimal Turing degree, there is a real code of α having the same Turing degree. □

Corollary 59. *There are continuum many minimal Turing degrees in the degree spectrum of each countable ordinal.*

Corollary 60. *There are continuum many minimal Turing degrees that hyper-arithmetically computes any Π^1_1 sets.*

We now explore the possibility of performing uniformly Turing cone avoiding in real codes. To make things easier we first consider uniform bT-cone avoiding. The degrees induced by the bounded Turing reducibility $\leq_{bT}$ are the bT-degrees. The following theorem shows how to avoid a bT-cone in an uniform way.

Theorem 61. *There is Turing functional with oracle $0'$, or a Δ^0_2 functional, Φ such that for any real code A of any infinite ordinal α and any incomputable set S, $\Phi(A, S)$ is a real code C such that S is not bounded Turing reducible to C, and more over if α is a limit ordinal then C also codes α and if α is a successor then C codes $\alpha + \omega$.*

Proof. Suppose $A \in \mathrm{WO}_{=\alpha}$. We construct C by finite extension s, will contain two part, a copy of A and above it a finite or ω-long increasing sequence of points. We reserve two disjoint computable copies of natural numbers for these two parts. During the construction we maintain a table of representations. Let C_n be the finite string after stage n.

In each $2n$ stage, we extend C_{2n-1} so that the points from the first copy of numbers in it form a copy of the the first n points of A.

In each $2n+1$ stage, we fix a string σ such that σ only adds a finite linear order using points from the second copy of ω above all points in C_{2n} and there is no $X \succ \sigma$ such that $\Phi_e(X) = S$ with the bounding function $\phi_{e'}$ where e and e' are the unique two number such that $\langle e, e' \rangle = n$. Such σ must exist and can be computed from $0'$ and S. (Let Y be a computable code that extends C_{2n} and adds a copy of ω using the second copy of numbers above all points of C_{2n}. Note that S can not be computed from the code Y by Φ_e with bounding function $\phi_{e'}$ because otherwise S would be computable.) That means there is a number k such that $S(k) \neq \Phi_e^{Y \upharpoonright \phi_{e'}(k)}(k)$. We choose the least such k and let σ be $Y \upharpoonright \phi_{e'}(k)$ if $\phi_{e'}(k) > |C_{2n}|$ and C_{2n} otherwise.

In every stage, we also make sure every point from the first copy of numbers is below every point from the second copy of numbers. In the end, C will code the same ordinal α if αs an infinite limit ordinal, and will code $\alpha + \omega$ if α is a limit ordinal, and $S \not\leq_{bT} C$ holds by construction. □

We do not know if similar uniform cone avoiding theorem can be proved for the Turing reducibility $\leq_T$.

Question 62. *Is there an arithmetical functional Φ such that for any real code A of any infinite limit ordinal α and any incomputable set S, $\Phi(A, S)$ is a real code C of the same ordinal such that S is not Turing reducible to C? If yes, what is lowest level in the arithmetical hierarchy to have such Φ?*

References

[AK00] C. J. Ash and J. Knight. *Computable structures and the hyperarithmetical hierarchy*, volume 144 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 2000.

[Can97] Georg Cantor. Beiträge zur Begründung der transfiniten Mengenlehre. *Math. Ann.*, 49(2):207–246, 1897.

[Coo73] S. B. Cooper. Minimal degrees and the jump operator. *J. Symbolic Logic*, 38:249–271, 1973.

[Dow98] R. G. Downey. Computability theory and linear orderings. In *Handbook of recursive mathematics, Vol. 2*, volume 139 of *Stud. Logic Found. Math.*, pages 823–976. North-Holland, Amsterdam, 1998.

[Jec03] Thomas Jech. *Set theory*. Springer Monographs in Mathematics. Springer-Verlag, Berlin, 2003. The third millennium edition, revised and expanded.

[Kle38] S. C. Kleene. On notation for ordinal numbers. *The Journal of Symbolic Logic*, 3(4):pp. 150–155, 1938.

[Med55] Yu. T. Medvedev. Degrees of difficulty of the mass problem. *Dokl. Akad. Nauk SSSR (N.S.)*, 104:501–504, 1955.

[Mos38] Andrzej Mostowski. *The Journal of Symbolic Logic*, 3(4):pp. 168–169, 1938.

[Ric81] Linda Jean Richter. Degrees of structures. *J. Symbolic Logic*, 46(4):723–731, 1981.

[Rog67] Hartley Rogers, Jr. *Theory of recursive functions and effective computability*. McGraw-Hill Book Co., New York, 1967.

[Sho59] J. R. Shoenfield. On degrees of unsolvability. *Ann. of Math. (2)*, 69:644–653, 1959.

[Soa87] Robert I. Soare. *Recursively enumerable sets and degrees*. Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987. A study of computable functions and computably generated sets.

[Soaar] Robert I. Soare. *Computability Theory and Applications: The Art of Classical Computability*. to appear.

[Sor96] Andrea Sorbi. The Medvedev lattice of degrees of difficulty. In *Computability, enumerability, unsolvability*, volume 224 of *London Math. Soc. Lecture Note Ser.*, pages 289–312. Cambridge Univ. Press, Cambridge, 1996.

[Spe55] Clifford Spector. Recursive well-orderings. *J. Symb. Logic*, 20:151–163, 1955.

Joel David Hamkins, The City University of New York, The Graduate Center, Mathematics Program, 365 Fifth Avenue, New York, NY 10016, & College of Staten Island of CUNY, 2800 Victory Blvd., Staten Island, NY 10314

E-mail address: jhamkins@gc.cuny.edu

URL: jdh.hamkins.org

please correct, Universiteit van Amsterdam, Institute of Logic, Language and Computation, P.O. Box 94242, 1090 GE Amsterdam, The Netherlands

E-mail address: z.li@uva.nl

URL: http://staff.science.uva.nl/~zhenhao/