ON ISOMORPHISM CLASSES OF COMPUTABLY ENUMERABLE EQUIVALENCE RELATIONS

URI ANDREWS AND SERIKZHAN A. BADAEV

Abstract. We examine how degrees of computably enumerable equivalence relations (ceers) under computable reduction break down into isomorphism classes. Two ceers are isomorphic if there is a computable permutation of \( \omega \) which reduces one to the other. As a method of focusing on non-trivial differences in isomorphism classes, we give special attention to weakly precomplete ceers. For any degree, we consider the number of isomorphism types contained in the degree and the number of isomorphism types of weakly precomplete ceers contained in the degree. We show that the number of isomorphism types must be 1 or \( \omega \), and it is 1 if and only if the ceer is self-full and has no computable classes. On the other hand, we show that the number of isomorphism types of weakly precomplete ceers contained in the degree can be any member of \( [0, \omega] \). In fact, for any \( n \in [0, \omega] \), there is a degree \( d \) and weakly precomplete ceers \( E_1, \ldots, E_n \) in \( d \) so that any ceer \( R \) in \( d \) is isomorphic to \( E_i \otimes D \) for some \( i \leq n \) and \( D \) a ceer with domain either finite or \( \omega \) comprised of finitely many computable classes. Thus, up to a trivial equivalence, the degree \( d \) splits into exactly \( n \) classes.

We conclude by answering some lingering open questions from the literature: Gao and Gerdes [12] define the collection of essentially FC ceers to be those which are reducible to a ceer all of whose classes are finite. They show that the index set of essentially FC ceers is \( \Pi_0^1 \)-hard, though the definition is \( \Sigma_0^1 \). We close the gap by showing that the index set is \( \Sigma_0^1 \)-complete. They also use index sets to show that there is a ceer all of whose classes are computable, but which is not essentially FC, and they ask for an explicit construction, which we provide.

Andrews and Sorbi [4] examined strong minimal covers of downwards-closed sets of degrees of ceers. We show that if \( (E_i) \) is a uniform c.e. sequence of non-universal ceers, then \( \{ \Theta_{i \leq j} E_i \mid j \in \omega \} \) has infinitely many incomparable strong minimal covers, which we use to answer some open questions from [4].

Lastly, we show that there exists an infinite antichain of weakly precomplete ceers.

1. Introduction

Computable reduction, a natural computability-theoretic analog of borel reduction and first introduced by Ershov [9, 10] as a computable representation for monomorphisms of numbered sets is defined by letting a binary relation \( R \) on \( \omega \) reduce to a binary relation \( S \) on \( \omega \) (written \( R \leq_c S \)) if there is a computable function \( f \) so that for every \( x, y \in \omega \), \( xRy \) if and only if \( f(x)Sf(y) \). The situation when \( R \) and \( S \) are equivalence relations, as in the borel theory, is of particular interest. In this paper, we continue the trend from [11 – 6] of examining the structure of the set of computably enumerable equivalence relations (ceers) under computable reduction. There has also been a study

2010 Mathematics Subject Classification. 03D25.

Key words and phrases. Computably enumerable equivalence relation; computable reducibility on equivalence relations; halting jump.

The first author’s research was partially supported by NSF grant DMS-160022 and by Grant 3952/GF4 of the Science Committee of the Republic of Kazakhstan. The second author’s research was partially supported by bina- tional NSF grant DMS-1600625 entitled “Collaboration in Computability” and by Grant AP05131579 of the Science Committee of the Republic of Kazakhstan.
of the relationship between ceers and the algebraic structures which have the ceer as its domain, see e.g., [11][13]-[16].

We also consider isomorphisms defined as follows: \( R \cong S \) if there is a computable function \( f \) which is a permutation of \( \omega \) so that \( xRy \) if and only if \( f(x)Sf(y) \). In this paper, we examine the question of how a \( \leq_c \)-degree splits into isomorphism classes. We show, in particular, that every degree contains either exactly 1 or infinitely many isomorphism classes, but there are degrees with “essentially” any finite number of isomorphism classes. Rigorously, this means that for any \( n \), there are \( \leq_c \)-classes which contain \( n \) ceers \( E_0, \ldots, E_{n-1} \) which are each non-isomorphic and have no computable classes so that any ceer \( R \) in the class is isomorphic to \( E_i \oplus D \) where \( D \) is a ceer (possibly on a finite domain) comprised of finitely many computable classes.

We also use the idea of a weakly precomplete ceer [6] as the idea of a ceer which is far from having any computable classes. Formally, a ceer \( E \) is weakly precomplete if it has no total computable diagonal function, i.e., there is no total computable \( f \) so that for every \( x \), \( xEf(x) \). Every two classes in a weakly precomplete ceer are computably inseparable, so such ceers are far from having computable classes. We examine some further properties of weakly precomplete ceers, but our main use is in constructing the ceers \( E \) above, which we make weakly pre-complete, so that we cannot have \( E_i \oplus D_i \cong E_j \oplus D_j \) where \( D_i \) and \( D_j \) are ceers with computable classes.

In sections 4, 5, and 6, we also answer some lingering questions in the literature regarding strong minimal covers of some natural subsets of ceers under \( \leq_c \), about the set of ceers which are \( \leq_c \) a ceer with only finite classes (the essentially FC ceers in [12]), and about antichains of weakly precomplete ceers.

2. Preliminaries

We begin with some standard definitions regarding ceers:

**Definition 2.1.** We let \( \text{Id}_n \) represent the ceer given by congruence modulo \( n \). Note that any ceer with exactly \( n \) equivalence classes is \( =_c \text{Id}_n \).

We let \( \text{Id} \) represent the ceer given by equality, i.e., \( x \text{Id} y \) if and only if \( x = y \).

**Definition 2.2** (The jump operation). For any ceer \( R \), we define \( R' \) to be the ceer given by \( xR'y \) if and only if \( x = y \) or \( \varphi_x(x)R\varphi_y(y) \) where both \( \varphi_x(x) \) and \( \varphi_y(y) \) converge.

The following observation will be helpful for building isomorphisms between ceers.

**Lemma 2.3.** If \( \varphi \) is a reduction of \( X \) to \( Y \) which is onto the classes of \( Y \), and both \( X \) and \( Y \) have no finite classes, then \( X \cong Y \).

*Proof.* We define a reduction \( f \) and a supplementary function \( g \) inductively in stages, so \( f = \bigcup_i f_i \). We ensure that each \( f_i \) is a partial reduction of \( X \) to \( Y \), and we ensure \( i \in \text{domain}(f_{i+1}) \cap \text{range}(f_{i+1}) \). We let \( f_0 = \emptyset \). If \( i \in \text{domain}(f_i) \), then we let \( g_i = f_i \). Otherwise, we enumerate \( [\varphi(i)]_Y \) until we see some member \( j \) of \( [\varphi(i)]_Y \setminus \text{range}(f_i) \). We then add \((i, j)\) to \( f_i \), to form \( g_i \). Now, if \( i \in \text{range}(g_i) \), then we let \( f_{i+1} = g_i \). Otherwise, we wait until we find some \( x \) so that \( \varphi(x) \in [i]_Y \) (since \( \varphi \) is onto the classes of \( Y \)), and we enumerate \( [x]_X \) until we find a member \( k \) which is not in \( \text{domain}(g_i) \). We then add \((k, i)\) to \( g_i \) to form \( f_{i+1} \). On classes, \( f = \varphi \), so \( f \) is also a reduction of \( X \) to \( Y \). By construction, \( f \) is a bijection. \( \square \)
**Definition 2.4.** If $E$ and $R$ are ceers, then $E \oplus R$ is the ceer defined by $x \in E \oplus R y$ if and only if either $x = 2n$ and $y = 2m$ and $n \not\equiv m$ or $x = 2n + 1$ and $y = 2m + 1$ and $n \not\equiv m$.

**Observation 2.5.** If $E$ and $R$ are incomparable, then the degree of $E \oplus R$ does not contain a weakly precomplete ceer.

Proof. Suppose $A$ is weakly precomplete and $A$ reduces to $E \oplus R$ via $f$, then $A$ reduces to either $E$ or $R$, since every pair of classes of $A$ are computably inseparable. That is, $\{x \mid f(x) \text{ is odd}\}$ and $\{x \mid f(x) \text{ is even}\}$ provides a separation of two classes, unless one is empty. Thus $f$ reduces $A$ to either $E$ or $R$. Since $E \oplus R$ does not reduce to $E$ or $R$, there can be no weakly precomplete $A$ in the same degree as $E \oplus R$. \hfill \square

**Observation 2.6.** The degrees of weakly precomplete ceers are not closed upwards.

Proof. Let $E$ be weakly precomplete and non-universal, which exists by [6]. Let $R$ be any ceer which is $\leq_c$-incomparable with $E$ (see e.g., [3, Theorem 2.1]). Then consider $E \oplus R$. This cannot be a weakly precomplete degree by Observation 2.5. \hfill \square

We also remind the reader of a useful definition which first appears in [5].

**Definition 2.7.** A ceer $E$ is self-full if $E \leq_c E \oplus \text{id}_1$. Equivalently (see [5]), and motivating the name, $E$ is self-full if whenever $\varphi$ is a $\leq_c$-reduction of $E$ to itself, $\varphi$ is onto the classes of $E$ (i.e. for every $j$, $\text{im}(\varphi) \cap [j]_E$ is non-empty).

We also note the following, which we will use to show hardness of an index set below:

**Observation 2.8.** For every binary $\Pi^0_3$-predicate $P(x, y)$ there exists a computable binary function $g$ so that $P(x, y) \iff W_{g(x, y)}$ is co-infinite.

Proof. By [18, Corollary 14-XVI], $\{x \mid W_x \text{ is co-finite}\}$ is $\Sigma^0_3$-complete set. Then, $\{< x, y > \mid P(x, y)\} \leq_1 \{x \mid W_x \text{ is co-infinite}\}$ by some computable function $f$. Define $g(x, y) = f(< x, y >)$. \hfill \square

### 3. Isomorphism-types inside degrees of ceers

Badaev and Sorbi [6] showed that there are infinitely many isomorphism types of universal weakly precomplete ceers. It is natural to ask whether there are non-universal weakly pre-complete ceers which are $\leq_c$-equivalent, but not isomorphic. We answer this question, introducing some techniques (especially the strategy for $D$-requirements) which will appear in the following theorems.

**Theorem 3.1.** There are non-isomorphic weakly precomplete ceers which are equivalent and non-universal.

Proof. We construct ceers $E$, $F$, and $X$ so that $E$ and $F$ are equivalent, non-isomorphic, weakly precomplete, and $X \not\equiv_c E$. During the construction, we will choose sequences of numbers $(a_i)_i \in \omega$
and \((b_i)_{i \in \omega}\), and we satisfy the following requirements:

\[ R_{E \rightarrow F} : \] For every pair \(i, j, i E j \Leftrightarrow a_i F a_j \)
\[ R_{F \rightarrow E} : \] For every pair \(i, j, i F j \Leftrightarrow b_i E b_j \)
\[ WP^i_E : \] If \(\varphi_i\) is a total function, then for some \(x, \varphi_i(x) E x \)
\[ WP^i_F : \] If \(\varphi_i\) is a total function, then for some \(x, \varphi_i(x) F x \)

\[ D_i : \] \(\varphi_i\) is not a reduction of \(E\) to \(F\)

\[ NU_i : \] \(\varphi_i\) is not an isomorphism of \(E\) to \(F\)

Note that the \(R\)-requirements are not subject to injury, but the others can be injured and re-initialized. We begin by describing the strategies for each requirement:

\(R_{E \rightarrow F}:\) We need to do 2 things to satisfy this requirement: choose \(a_k\), and collapse to maintain consistency. For the least \(k\) where \(a_k\) is not defined, we choose \(a_k\) to be a fresh number (i.e. larger than any number mentioned in the construction). We collapse \(a_i\) to \(a_j\) in \(F\) if we see \(i E j\). We will ensure – see Corollary 3.6 – that no other requirement collapses a pair \(a_i\) and \(a_j\) (i.e. \(a_i F a_j\) only happens if we already see \(i E j\)).

\(R_{F \rightarrow E}:\) This is symmetric.

\(WP^i_E: \) Choose a witness \(x\) to be fresh. Wait for \(\varphi_i(x)\) to converge. Then \(E\)-collapse \(x\) with \(\varphi_i(x)\). After this collapse, we no longer consider the requirement active.

\(WP^i_F: \) This is symmetric.

\(D_i: \) Let \(x, x', z\) be fresh. We wait for a stage \(t\) where \(\varphi_i(x)\) and \(\varphi_i(x')\) converge and \(\varphi_i(y) = z\) for some \(y\). If \(\varphi_i(x) F^t \varphi_i(x')\), then we do nothing further (we will see that, unless injured, \(xE x'\)) and no longer consider the requirement active. Otherwise, by possibly reversing \(x\) and \(x'\), we may assume that \(\varphi_i(x) F^t z\). Let \(w = \varphi_i(x)\). We collapse \(x E y\). We will see below that (unless injured) \(zF^t w\). After this collapse, we no longer consider the requirement active.

\(NU_i: \) Take two fresh numbers \(x, y\). Wait for \(\varphi_i(x)\) and \(\varphi_i(y)\) to converge. If \(\varphi_i(x) F \varphi_i(y)\), then collapse \(x X y\). We will see below that reinitializing lower-priority requirements will suffice to guarantee that \(\varphi_i(x) E \varphi_i(y)\) remains true.

**Construction.** We fix some priority ordering in order type \(\omega\) of the \(WP\), \(D\) and \(NU\)-requirements. As they are not subject to injury, we do not include \(R_{E \rightarrow F}\) and \(R_{F \rightarrow E}\)-requirements in our priority order. We deal with each of \(D\), \(WP\), and \(NU\)-requirements at infinitely many stages, one at every stage \(s > 0\) of the construction. And we deal with the \(R\)-requirements at each stage of the construction. \(WP^i_E\) and \(WP^i_F\)-requirements can have a parameter \(x\). \(D_i\)-requirements can have parameters \(x, x',\) and \(z\). \(NU_i\)-requirements can have parameters \(x\) and \(y\). When a requirement is initialized, each parameter is set to be undefined and the requirement is set to be active. We say that a requirement requires attention if any of its parameters are undefined, or if it is an active \(WP^i\)-requirement and \(\varphi_i(x)\) has converged, or it is an active \(D_i\)-requirement and \(\varphi_i(x), \varphi_i(x')\) converge and some \(\varphi_i(y) = z\), or if it is an active \(NU_i\)-requirement and \(\varphi_i(x)\) and \(\varphi_i(y)\) have both converged.
When a strategy for a \( WP, D \), or NU-requirement acts, it re-initializes all lower-priority \( D, WP \), or NU-requirements. Any re-initialized requirement becomes active immediately.

**Stage 0.** Initialize each \( WP \)-, \( D \)-, and NU-requirement. Set \( E^0 = Id, F^0 = Id \).

**Stage \( s+1 \).** Let \( s = (s_0, s_1) \) and let \( Q \) be the strategy with priority equal to \( s_0 \). If \( Q \) does not require attention, then end the stage. Otherwise, we say that \( Q \) acts, thus reinitializing all lower-priority requirements. We consider cases depending on which type of requirement \( Q \) is:

**Case 1.** Suppose \( Q \) is a \( WP^i \)-requirement. If the parameter \( x \) is not yet defined, define it to be a fresh number (in particular \( [x]_E^s = \{x\} \) and \( x \) is not a parameter of any other requirement). Otherwise, we have \( x \) is defined and \( \varphi_i(x) \) converges. Then we \( E \)-collapse \( x \) with \( \varphi_i(x) \) and declare \( WP^i_E \) to be inactive.

**Case 2.** Suppose \( Q \) is a \( WP^i \)-requirement. We act exactly as in case 1, but with \( F \)-collapse.

**Case 3.** Suppose \( Q \) is a \( D_i \)-requirement. If \( x, x' \) and \( z \) are undefined, then select them to be distinct fresh numbers. Otherwise, we have \( \varphi_i(x), \varphi_i(x') \) converged and \( \varphi_i(y) = z \) for some \( y \). If \( \varphi_i(x)F\varphi_i(x') \), we declare \( D_i \) to be inactive and do nothing else. Otherwise, by possibly switching the roles of \( x \) and \( x' \), we may assume that \( \varphi_i(x)\varphi_i^+ \). We \( E \)-collapse \( x \) with \( y \) and declare the requirement \( D_i \) to be inactive.

**Case 4.** Suppose \( Q \) is a NU_i-requirement. If \( x \) and \( y \) are not defined, then we select them to be fresh numbers. Otherwise, we have \( \varphi_i(x) \) and \( \varphi_i(y) \) both converged. If \( \varphi_i(x)E\varphi_i(y) \), then we declare \( NU_i \) to be inactive. Otherwise, we \( X \)-collapse \( x \) and \( y \) and declare \( NU_i \) to be inactive.

In any case, we finish the stage with the following:

**Coding Step:** Define \( a_s \) and \( b_s \) to be fresh numbers. Lastly, if we have \( E \)-collapsed \( x \) and \( y \), and \( a_x \) and \( a_y \) are defined, then \( F \)-collapse \( a_x \) and \( a_y \). Similarly, if we have \( F \)-collapsed \( x \) and \( y \), and \( b_x \) and \( b_y \) are defined, then \( E \)-collapse \( b_x \) and \( b_y \). We apply this as many times as necessary, but at stage \( s+1 \), we have only defined finitely many values of \( a_x \) and \( b_x \), so this only causes finitely many collapses.

**Verification.** We proceed through a sequence of lemmas to show that all requirements are satisfied.

**Definition 3.2.** Let \( Q \) be a WP, D or NU-requirement. We say that \( x \) is a \( Q \)-number at stage \( s \) if \( x \) is a parameter of an active \( Q \)-requirement.

We say that \( x \) is an \( R_{E \rightarrow F} \)-number at stage \( s \) if it is defined to be \( a_i \) for some \( i \) so that \( i \) is the least member of \([i]_E \) at stage \( s \).

We say that \( x \) is an \( R_{F \rightarrow E} \)-number at stage \( s \) if it is defined to be \( b_i \) for some \( i \) so that \( i \) is the least member of \([i]_F \) at stage \( s \).

In each case, we say that the number \( x \) is active at stage \( s \).

**Lemma 3.3.** Let \( x \) and \( y \) be distinct active numbers at stage \( s \). Then \( xE^s y \) and \( xF^s y \).

**Proof.** This is clearly true at stage 0. Suppose \( s + 1 \) is the least stage at which this lemma fails. Let \( x, Q_1, y, \) and \( Q_2 \) witness this. Let us consider the action at stage \( s \) which brought about this situation. At stage \( s \), we must have done more than just defining new parameters, because all new parameters are chosen to be fresh. In particular, if \( z \) is fresh, then \([z]_E^s = [z]_F^s = \{z\} \), so it cannot contribute to violating our lemma.
There are two parts of the construction at stage \( s \): The action in each of the 4 cases, and then the coding step. We verify that after each of these actions, we have not violated our lemma.

In each of the 4 cases where we can cause a collapse, we have a requirement \( Q \)-which collapses one of its parameters \( z \) to some other element \( w \). We then declare \( Q \) to be inactive. By inductive hypothesis, \([z]_E \) and \([z]_F \) contains only one active number, namely \( z \). Thus, since \( z \) is not active at stage \( s + 1 \) since \( Q \) becomes inactive, we have added no new active numbers to \([w]_E \) or \([w]_F \).

Lastly, we have to check that our collapses during the coding step do not cause us to violate this lemma. These are of the form of \( F \)-collapsing \( a_i \) with \( a_j \) if we have \( E \)-collapsed \( i \) with \( j \) or of the form of \( E \)-collapsing \( b_i \) with \( b_j \) if we have \( F \)-collapsed \( i \) with \( j \). We consider the former case as the latter case is the same. We can assume that prior to \( E \)-collapsing \( i \) with \( j \), both were least in their \( E \)-classes. Thus, both \( a_i \) and \( a_j \) were active. It follows that \([a_i]_F \) and \([a_j]_F \) had only one active element, namely \( a_i \) and \( a_j \). But since \( i \) and \( j \) have collapsed, one has stopped being active. So the newly formed class \([a_i]_F \cup [a_j]_F \) still contains only one active element.

\[ \square \]

**Lemma 3.4.** Let \( x = b_i \). Then for every \( s > i \) there exists \( a_j \leq i \) so that \( xE^sb_j \) and \( b_j \) is active at stage \( s \).

Let \( x = a_i \). Then for every \( s > i \) there exists \( a_j \leq i \) so that \( xF^sa_j \) and \( b_j \) is active at stage \( s \).

**Proof.** We prove the first claim. Let \( j \) be the least number in \([i]_F \). Then \( j \leq i \) and \( b_j \) is active. By the coding step of our construction, \( iF^s j \) implies \( b_iE^sb_j \).

**Lemma 3.5.** Let \( x \) be a number mentioned before stage \( s \). Suppose that \( x \) is not \( E^s \)-equivalent to any active number at stage \( s \). Then at all stages \( t > s \), \( x \) is not \( E^t \)-equivalent to any active number at stage \( t \).

Similarly for \( F \)-equivalence.

**Proof.** Suppose otherwise, and consider the first stage \( t > s \) at which \( x \) becomes \( E^t \)-equivalent to an active number at stage \( t \). This cannot be caused by an assignment of parameters, since all parameters are assigned to be fresh. By the same analysis as in Lemma 3.3 any active \( z \) which is collapsed with \( x \) must simultaneously become inactive. Similarly, this cannot be caused by collapsing \( b_i \) with \( b_j \) for the sake of coding, because \( x \) cannot already be in \([b_i]_E \) or \([b_j]_E \), since these each contain active members.

**Corollary 3.6.** For every \( i, j < s \), \( iE^s j \) if and only if \( a_iF^s a_j \). For every \( k, l < s \), \( kF^sl \) if and only if \( b_kE^sb_l \).

**Proof.** We prove only the first claim as the second is symmetric. By the coding step, \( iE^s j \) implies that \( a_iF^s a_j \). To see the reverse, suppose that \( iE^s j \) and let \( i_0 \) be least in \([i]_E \) and \( j_0 \) be least in \([j]_E \). It follows by the coding step that \( a_{i_0}F^s a_i \) and \( a_{j_0}F^s a_j \). Then \( a_{i_0} \) and \( a_{j_0} \) are both active numbers at stage \( s \). It follows by Lemma 3.3 that they cannot be \( F^s \)-equivalent. Thus \( a_iF^s a_j \).

It follows that the \( R \)-requirements are satisfied.

**Lemma 3.7.** Every requirement is re-initialized only finitely often.

**Proof.** Straightforward by induction in priority of the requirements.
Lemma 3.8. Suppose that \( x \) and \( y \) are numbers considered before stage \( s \) and \( x E^s y \). Suppose that \( Q \) is a requirement which is deactivated at stage \( s \) (thus all lower-priority requirements are reinitialized at stage \( s \)). Suppose further that no requirement of higher priority than \( Q \) acts after stage \( s \). Then \( x E y \). Similarly for \( F \).

Proof. Case 1: Neither \( x \) nor \( y \) are \( E^s \)-equivalent to any active number. Then by Lemma 3.5, this is true at every \( t > s \). Then at any stage \( t > s \) we cannot collapse \( x \) with \( y \) because neither are equivalent to any active numbers.

Case 2: Suppose that either \( x \) or \( y \) is \( E^s \)-equivalent to a \( \hat{Q} \)-number at stage \( s \) for \( \hat{Q} \) a \( WP \), \( D \) or \( NU \)-requirement of higher priority than \( Q \). WLOG, we suppose this is true of \( x \). Suppose towards a contradiction that \( x E y \). Let \( t > s \) be the stage at which we cause this collapse. Since \( \hat{Q} \) does not act after stage \( s \), we know that \( x \) is also \( E^t \)-equivalent to a \( \hat{Q} \)-number at stage \( t \), and thus cannot be \( E^t \)-equivalent to any other active number by Lemma 3.3. Thus, the collapse must be caused by an active number \( E^t \)-equivalent to \( y \).

Case 2a: Suppose that \( y \) is not \( E^s \)-equivalent to an active number at stage \( s \). Then by lemma 3.5, this is true at stage \( t \) also, so the collapse cannot occur at stage \( t \).

Case 2b: Suppose that \( y \) is \( E^s \)-equivalent to \( b_j \) for some \( j \). Then by Lemma 3.4, \( y \) is \( E^t \)-equivalent to some active \( b_k \) for some \( k \). Thus, it is not equivalent to a \( \hat{Q} \)-number at stage \( t \) for \( \hat{Q} \) any \( WP \), \( D \) or \( NU \)-requirement. Since \( x \) cannot be equivalent to any \( b_l \) (since Lemma 3.4 shows that it would then be \( E^t \)-equivalent to two active numbers contradicting Lemma 3.3), the collapse can also not occur due to the coding.

Case 2c: Suppose that \( y \) is also \( E^s \)-equivalent to a \( \hat{Q}' \)-number at stage \( s \) for \( \hat{Q}' \) a \( WP \), \( D \) or \( NU \)-requirement of higher priority than \( Q \). Then, \( y \) is also \( E^t \)-equivalent to a \( \hat{Q}' \)-number at stage \( t \), and thus cannot be \( E^t \)-equivalent to any other active number by Lemma 3.3. Thus, since neither \( \hat{Q}' \) nor \( \hat{Q} \) can act at stage \( t \), we cannot cause the collapse at stage \( t \).

Case 3: Suppose that \( x E^s b_i \) and \( y \) is not \( E^s \)-equivalent to any active number. Then Lemma 3.4 shows that \( [x]_{E^t} \) contains an active number for every \( t > s \) whereas Lemma 3.5 shows that \( y \) is never \( E^t \)-equivalent to an active number, so \( x \) and \( y \) can not be \( E^t \)-equivalent for any \( t > s \).

Case 4: We have \( x E^s b_i \) and \( y E^s b_j \) for some \( i,j \). Then the only cause of the collapse of \( x E y \) is due to the coding step, since neither can ever be \( E^t \)-equivalent to any other active number. But then we can consider why we collapse \( iF j \). By cases 3, the only possibility is that this, in turn was due to Case 4, namely due to a coding step. But the coding step at any given stage is finite and originates in a collapse for a \( WP \), \( D \), or \( NU \)-requirement, which we have ruled out in the cases above.

Lemma 3.9. \( WP_E^i \) and \( WP_F^i \)-requirements are satisfied.

Proof. Consider the last time the requirement is reinitialized. When it next chooses its witness \( x \), this choice is permanent. If \( \varphi_i(x) \) does not converge, then the requirement is satisfied. Otherwise, once it converges, the requirement will act (since no higher priority requirement can act) by collapsing \( x \) with \( \varphi_i(x) \) satisfying the requirement.

Lemma 3.10. The \( D_1 \)-requirement is satisfied.
Proof. Consider the last time the requirement is reinitialized. When it next chooses its witnesses $x, x', z,$ this choice is permanent. If $ϕ_i(x)$ or $ϕ_i(x')$ do not converge or no $ϕ_i(y)$ converges to equal $z$, then $ϕ_i$ is not a bijection and the requirement is satisfied. Now, suppose $ϕ_i(x), ϕ_i(x')$ converge and $ϕ_i(y)$ converges to equal $z$, and let $t$ be the first stage after this when this requirement is next considered by the construction. If $ϕ_i(x)F^tϕ_i(x')$, the requirement does nothing. As $x$ and $x'$ were active at stage $t$, they were not $E^♯$-equivalent. By Lemma 3.8, we see $xE x'$. Otherwise, (possibly after reversing $x$ and $x'$), we have $ϕ_i(x) = wF^♯z$. Then we $E$-collapsed $x$ with $y$. It suffices to show that $wFz$. Since $z$ is not equivalent to any $a_i$, $w$ and $z$ do not collapse at stage $t$. Since $w$ and $z$ are certainly considered at stage $t$, the $D_i$-requirement becomes inactive at stage $t$, and we have supposed that no higher priority requirement acts after stage $t$ (as this would reinitialize the $D_i$-requirement), Lemma 3.8 guarantees that $wFz$.

Lemma 3.11. The requirement $NU_i$ is satisfied.

Proof. Consider the last time that $NU_i$ is reinitialized. When it next picks its witnesses $x, y$, this is permanent. If $ϕ_i(x)$ or $ϕ_i(y)$ never converge, then the requirement is satisfied. Otherwise, consider the next stage $s$ where $NU_i$ acts. If $ϕ_i(x)E^sϕ_i(y)$ then simply not $X$-collapsing $x$ and $y$ guarantees that the requirement is satisfied. So, we must consider the case that $ϕ_i(x)E^♯ϕ_i(y)$ and we must show that $ϕ_i(x)Eϕ_i(y)$. This follows immediately by Lemma 3.8.

Thus, every requirement eventually succeeds, and we have built ceers $E$ and $F$ as needed.

Theorem 3.12. There are non-universal weakly precomplete ceers $E_i$ for $i ∈ ω$ so that they are equivalent and pairwise non-isomorphic.

Proof. This is the same argument as the previous theorem with no new complications. We construct infinitely many ceers, but the requirements each mention at most two ceers and are handled with strategies identical to the previous argument.

Definition 3.13. For a ceer $E$, we define $N(E)$ to be the number of isomorphism types inside the $≤_c$-degree of $E$. We define $N^*(E)$ to be the number of isomorphism types of weakly precomplete ceers inside the $≤_c$-degree of $E$.

Theorem 3.14. $N(E) = 1$ if and only if $E$ is self-full and has no computable classes. Otherwise, $N(E) = ω$.

Proof. We prove the theorem in cases:

Lemma 3.15. If $E$ is self-full and has no computable classes then $N(E) = 1$.

Proof. Suppose $X ≡_c E$. Then consider the reductions $E ≤_c X ≤_c E$. Since $E$ is self-full, it follows that the composed reduction is onto the classes of $E$, therefore, the reduction $E ≤_c X$ is onto the classes of $X$. Thus, $X$ cannot have any computable classes either, so all of the classes of $X$ are infinite. Thus $E ≡ X$ by Lemma 2.3.

Lemma 3.16. If $E$ has no computable classes and is non-self-full, then $N(E) = ω$. 

Proof. Suppose $X ≡_c E$. Then consider the reductions $E ≤_c X ≤_c E$. Since $E$ is self-full, it follows that the composed reduction is onto the classes of $E$, therefore, the reduction $E ≤_c X$ is onto the classes of $X$. Thus, $X$ cannot have any computable classes either, so all of the classes of $X$ are infinite. Thus $E ≡ X$ by Lemma 2.3.
Proof. Consider the set of equivalence relations $E \oplus \text{Id}_n$ for various $n$. These are all non-isomorphic because $E \oplus \text{Id}_n$ has exactly $n$ computable classes, yet they are all $\leq_c$-equivalent by non-self-fullness of $E$. □

Lemma 3.17. If $E$ has a computable class, then $N(E) = \omega$.

Proof. Suppose, towards a contradiction, that $N(E) = k$ and let $E_0, E_1, \ldots, E_{k-1}$ be representatives of every isomorphism type in the degree of $E$.

Claim 3.18. For some $i < k$, there are infinitely many $m$ so that $E_i$ has a finite class of size $m$.

Proof. For each $m \in \omega$, let $R_m$ be the ceer formed by replacing a computable class in $E$ by a class of size $m$. Let $F(m)$ be the $i < k$ so that $R_m \cong E_i$. By the pigeonhole principle, there is an infinite set of $m$ on which $F$ is constant. □

Without loss of generality, we assume $E_0$ has this property.

Claim 3.19. For all positive $n \in \omega$, there is an $E_i$ with infinitely many finite classes, but none of size $\leq n$.

Proof. For each $n > 0$, consider the ceer $Q_n$ formed by taking $E_0$ and fattening every point by $n$ numbers. That is, we say $xQ_n y$ if and only if $[\frac{x}{n}]E_0 [\frac{y}{n}]$. □

For each $i < k$, let $l_i \in \omega$ be greater than the size of the smallest finite class of $E_i$, if there is one, and 1 otherwise. Applying the previous claim with $n = \Sigma_{i<k} l_i$ gives $Q_n$, a new isomorphism type in the degree of $E$.

□

It may seem like a very weak argument that $N(E)$ is infinite in Lemmas 3.16 and 3.17. One might argue that appending finitely many computable classes to $E$ may yield a new isomorphism type, but it does not give a substantively different isomorphism type. We will see that for some ceers $E$, the only way to produce other isomorphism types in the same degree is to append finitely many computable classes to $E$.

Corollary 3.20. There is a weakly precomplete ceer $E$ so that for any $X$, $X \equiv_c E$ implies $X \cong E$.

Proof. It is shown in [6] (or see Theorem 6.3) that there are weakly precomplete ceers $E$ which are dark, i.e., $\text{Id} \not\leq_c E$. It is shown in [5] that all dark ceers are self-full. Lastly, since the classes of weakly precomplete ceers are computably inseparable, no class can be computable. So, a dark weakly precomplete ceer $E$ has $N(E) = 1$ by Lemma 3.15. □

Theorem 3.21. The range of $N^*$ is $[0, \omega]$.

Proof. The fact that 0, 1 and $\omega$ are in the range follows from Observation 2.5, Lemma 3.15 and Theorem 3.12. For $n \in (1, \omega)$, we give the following construction:

We build ceers $E_0, \ldots, E_n$, and we will make $E_0 = E_n$. Towards this, whenever we give instructions to $E_0$-collapse some pair, it is understood that we simultaneously $E_n$-collapse and vice-versa. We also build functions $\pi_i$ for $i = 1, \ldots, n$ so that $\pi_i$ reduces $E_{i-1}$ to $E_i$. Thus $\pi = \pi_n \circ \pi_{n-1} \circ \cdots \circ \pi_1$.
is a reduction of $E_0$ to itself. We attempt to ensure that the set of weakly precomplete ceers in the degree of $E_0$ is exactly $\{E_0, \ldots, E_{n-1}\}$, and these are pairwise non-isomorphic, thus $N^*(E_0) = n$. We build these ceers with the following requirements (where we consider $i, i' < n$ distinct and any $j, k \in \omega$):

\[ C^i : \text{Pick numbers } a_j^{i+1} \text{ for each } j \text{ so that } jE_i k \text{ if and only if } a_j^{i+1} E_{i+1} a_k^{i+1} \]

\[ WP_j^i : \text{If } \varphi_j \text{ is a total function, then it has an } E_i \text{-fixed point.} \]

\[ D_j^{i,i'} : \varphi_j \text{ is not an isomorphism from } E_i \text{ to } E_{i'}. \]

\[ T_{j,k}^i : \text{If } W_j \text{ intersects infinitely many } E_i \text{-classes which do not contain any element of the form } a_j^i, \text{ then } W_j \text{ intersects } [k]_{E_i}. \]

\[ S_j : \text{If } \varphi_j \text{ is a reduction of } E_0 \text{ to } E_0, \text{ then for some } k, [\text{im}(\varphi_j)]_{E_0} \cap ([\text{im}(\pi^k)]_{E_0} \setminus [\text{im}(\pi^{k+1})]_{E_0}) \text{ contains infinitely many classes.} \]

See Lemmas 3.32 and 3.33 for why the requirements, especially $T_{j,k}^i$ and $S_j$, suffice for the theorem. Intuitively, $\pi$ gives us a nested layering of $E_0$ by smaller copies of itself, and $S_j$ shows that any ceer $R$ equivalent to $E_0$ must in its reduction to $E_0$ intersect one annulus infinitely. We then stratify that annulus in terms of the $E_i$’s and see that one of these strata must be hit infinitely. Then $T_{j,k}^i$ shows that the entirety of this copy of $E_i$ must be hit, which is enough for us to analyze the ceer $R$.

We enumerate all $WP, D, T$ and $S$-requirements in order type $\omega$ as $Q_0 < Q_1 \cdots$.

At this point, the strategies for $C^i$, $WP_j^i$, and $D_j^{i,i'}$ should be familiar. We highlight the strategies for $T_{j,k}^i$ and $S_j$ and their conflict.

\[ T_{j,k}^i \text{-strategy:} \text{ Wait for } W_j \text{ to enumerate a number } x \text{ which is not } E_i \text{-equivalent to any number mentioned by a higher-priority requirement and is also not } E_i \text{-equivalent to any element of the form } a_j^i. \text{ Then collapse } kE_i x. \]

\[ S_j \text{-strategy:} \text{ Throughout the description, we let } l \text{ be maximal so that } y^l \text{ is defined.} \]

Step 1: Pick new $y^0, y^1$, and keep $y^0E_0 y^1$ and that neither $y^0$ nor $y^1$ will ever be equivalent to an element of the form $a_k^n$ for any $k$ (this will be automatic via Lemma 3.25, which is analogous to Lemma 3.3). Wait for $\varphi_j(y^0)$ and $\varphi_j(y^1)$ to converge. Once this happens, go to step 2.

Step 2: If for some $k$ we have $\{[\varphi_j(y^0)]_{E_0}, [\varphi_j(y^1)]_{E_0} \} = \{[\pi^k y^0]_{E_0}, [\pi^k y^1]_{E_0} \}$, then go to step 3. Otherwise, we have two cases: If $\varphi_j(y^0) E_0 \varphi_j(y^1)$ already, then we simply maintain that $y^0E_0 y^1$ and do nothing. Otherwise, we collapse $y^0$ with $y^l$, and we will not be forced to collapse $\varphi_j(y^0)$ with $\varphi_j(y^1)$. We reinitialize lower priority requirements to ensure that they will not cause $\varphi_j(y^0)$ and $\varphi_j(y^1)$ to $E_0$-collapse. We do nothing further.

Step 3: Choose a new number $y$ that is not equivalent to any element of the form $a_k^n$ and assign this to be $y^{l+1}$. Wait for $\varphi_j(y^{l+1})$ to converge, then go back to step 2 (with the newly increased value of $l$).
The possible outcomes of one $S_j$ strategy are either infinite cycling through step 2 and step 3, or it gets stuck in step 1, 2, or 3. If it cycles through steps 2 and 3 infinitely often, this will force that for a single $k$, every $m$ satisfies $\varphi_j(y^m)E_0\pi^k y^m$ (see Lemma 3.30). Further, since $y^mE_0\pi(d)$ for every $d$, we have that $\pi^{k\cdot}y^mE_0\pi^{k+1}\cdot d$ for every $d$. Thus this gives that $[\text{im}(\varphi_j)]E_0 \cap [\text{im}(\pi^k)]E_0 \setminus [\text{im}(\pi^{k+1})]E_0$ is infinite as needed. If it gets stuck in step 1 or step 3, then $\varphi_j$ is not total, and if it gets stuck in step 2, then we diagonalize ensuring that $\varphi_j$ is not a reduction of $E_0$ to itself.

Now, note that $S_j$ has infinitely many parameters if it cycles through steps 2 and 3 infinitely often, which is inconsistent with $T_{j,k}^i$-strategies (it causes no problems to $D$ or $WP$-strategies as these can only cause a collapse involving at least one new number, whereas $T_{j,k}^i$-strategies use numbers chosen by $W_j$). In particular, if $W_j$ enumerates $\{y^m \mid m \in \omega\}$ chosen by a higher-priority $S_j$-requirement, it would not be able to be satisfied. This problem is fixed by allowing a high enough priority $T_{j,k}^i$-requirement, say it is requirement $Q_p$ to only respect values $y^m$ for $m < p$. In other words, the lower-priority $T$-requirement may steal the $S$-requirement’s parameter $y^k$ for $k \geq p$. Still, only finitely many $T$-strategies may steal this parameter, so if $S$ cycles through 2 and 3 infinitely often, it will eventually define each $y^m$.

The remainder of the proof is handled via the usual priority machinery.

**Construction.** We build equivalence relations $E_0^s, E_1^s, \ldots, E_{n-1}^s$ as approximations to the equivalence relations $E_0, E_1, \ldots, E_{n-1}$. $WP$-requirements can have a parameter $x$. $D$-requirements can have parameters $x, x', \ldots, x_l$ and $z$. $T$-requirements can have a parameter $r$. $S$-requirements can have, as parameters, a finite sequence of numbers $y^0, \ldots, y^l$.

We say that a $WP$-requirement demands attention if it is active and either $x$ is undefined or $\varphi_j(x)$ has converged. We say that a $D$-requirement requires attention if it is active and either its parameters are undefined or $\varphi_j(x), \varphi_j(x')$ have converged and some $\varphi_j(y) = z$. We say a $T_{j,k}^i$-requirement requires attention if it is active and either $r$ is undefined or some number $x$ has been enumerated into $W_j$ so that $x$ is not $E_0^s$-equivalent to any $a^i_l$ or any number less than $r$. We say that an $S_j$-requirement requires attention if it is active and either $y^0$ is undefined or, for $l$ being greatest so that $y^l$ is defined, we have both $\varphi_j(y^0)$ and $\varphi_j(y^l)$ converged and either $\{[\varphi_j(y^0)]E_0, [\varphi_j(y^l)]E_0\} = \{[\pi^k y^0]E_0, [\pi^k y^l]E_0\}$ for some $k$ or $\varphi_j(y^0)E_0[\varphi_j(y^l)]$.

When a requirement is initialized, each parameter is set to be undefined and the requirement is set to be active. For a $WP$, $D$, $T$ or $S$ requirement $Q$, let $\#(Q)$ be the number $m$ so that the requirement is $Q_m$. At stage $s+1 = \langle m, k \rangle$, we consider the requirement $Q_m$, thus each requirement is considered infinitely often.

**Stage 0.** Initialize all requirements. For every $i < n$, set $E_i^0 = \text{Id}$.

**Stage** $s + 1$. Denote by $Q$ the requirement that we consider at stage $s + 1$. If $Q$ does not require attention then go to the next stage, otherwise we execute the strategy for $Q$ below, followed by the coding step:

**Case 1.** $Q$ is the $WP_{j^i}^i$-requirement for some $i < n$ and $j \in \omega$.

**Case 1.1.** If a parameter $x$ for $WP_{j^i}^i$ is not defined then choose $x$ to be a fresh number. Reinitialize all lower priority requirements.
Case 1.2. If $x$ is the parameter for $WP^i_j$ and $\varphi_j^i(x)$ has converged, then collapse $xE^i_{i+1} \varphi_j(x)$, reinitialize all lower-priority requirements, and declare the $WP^i_j$-requirement to be inactive.

Case 2. $Q$ is the $D^i_{j,i'}$-requirement with $i, i' < n, i \neq i'$ and $j \in \omega$.

Case 2.1. If no parameters for $D^i_{j,i'}$ are defined then pick three fresh numbers $x, x', z$ and set them to be the parameters for $D^i_{j,i'}$. Reinitialize all lower-priority requirements.

Case 2.2. If the parameters $x, x', z$ for $D^i_{j,i'}$ are defined and $\varphi_j^i(x), \varphi_j^i(x')$ are converged, and for some $y, \varphi_j^i(y) = z$, then: If $\varphi_j(x)E^i_{i'} \varphi_j(x')$, we declare $D_i$ to be inactive and do nothing else. Otherwise, by possibly switching the roles of $x$ and $x'$, we may assume that $\varphi_i(x)E^i_{i'}z$.

We $E_i$-collapse $x$ with $y$ and declare the requirement $D^i_{j,i'}$ to be inactive. We re-initialize all lower-priority requirements.

Case 3. $Q$ is the $T^i_{j,k}$-requirement with $i < n$ and $j, k \in \omega$.

Case 3.1. If $k \in [W^e_j]_{E_i,s}$ then declare $T^i_{e,k}$-requirement to be inactive.

Case 3.2. If $k \notin [W^e_j]_{E_i,s}$ and $r$ is not defined, then choose $r$ to be a fresh number. Reinitialize all lower-priority requirements.

Case 3.3. If there is an $x$ in $W_j$ so that $x$ is not $E_i$-equivalent to any number less than $r$ and also not $E_i$-equivalent to $a^l_i$ for any $l$, then $E_i$-collapse $x$ with $k$. We declare $Q$ to be inactive and reinitialize all lower-priority requirements. If this number $x$ is already $E_i$-equivalent to a higher-priority $S_j$-requirement’s parameter $y^m$, then we redefine the $S_j$-requirement’s parameters $y^{m'}$ for every $m' \geq m$. (In Lemma 3.22, we will see that this can only happen if $m > \#(Q)$.)

Case 4. $Q$ is the $S_j$-requirement. If $S_j$ does not require attention, then do nothing. Otherwise:

Case 4.1. The parameter $y^0$ is not defined. Then choose $y^0$ and $y^1$ to be fresh numbers. Reinitialize all lower priority requirements.

If the parameter $y^0$ is defined, then let $l$ be largest so that the parameter $y^{l}$ is defined.

Case 4.2. We have $\varphi_j(y^0)$ and $\varphi_j(y^l)$ both converged and $\{[\varphi_j(y^0)]_{E_0}, [\varphi_j(y^l)]_{E_3} \} = \{[\pi^k y^0]_{E_0}, [\pi^k y^l]_{E_3} \}$ for some $k$. In this case, define the parameter $y^{l+1}$ to be fresh and initialize all lower priority requirements $\mathcal{R}$ so that $\#(\mathcal{R}) \geq l$.

Case 4.3. We have $\varphi_j(y^0)$ and $\varphi_j(y^l)$ both converged and $\{[\varphi_j(y^0)]_{E_0}, [\varphi_j(y^l)]_{E_3} \} \neq \{[\pi^k y^0]_{E_0}, [\pi^k y^l]_{E_3} \}$ for any $k$, and $\varphi_j(y^0)E_0^r \varphi_j(y^l)$. Then $E_0$-collapse $y^0$ and $y^l$ and reinitialize all lower priority requirements. Declare $S_j$ to be inactive.

Coding Step: For each $i = 1, \ldots, n$, choose distinct fresh numbers to be $a^i_j$. For each $i < n$, if we have $E_i$-collapsed $j$ with $k$, and $a^i_{j+1}$ and $a^i_{k+1}$ are defined, then $E_{i+1}$-collapse $a^i_{j+1}$ and $a^i_{k+1}$. Note that since we have only finitely many $a^i_{j+1}$-values defined, this causes only finitely many collapses.

Verification.

Lemma 3.22. If a $T$-requirement undefines an $S$-requirement’s parameter $y^m$, then $\#(T) < m$.
Proof. It must be that $x E_i y^m$, but $x E_z^i z$ for every $z \leq r$. Thus $y^m > r$. In particular, $y^m$ was chosen to be a parameter after the $T$-requirement assigned the parameter $r$. So when $y^m$ was assigned in Case 4.2, the $T$-requirement was not reinitialized, thus $\#(T) < m$. □

Lemma 3.23. Each requirement is reinitialized only finitely often.

Proof. Suppose towards a contradiction that $Q$ is the highest priority requirement reinitialized infinitely often. There must be a single higher priority requirement $R$ which reinitializes $Q$ infinitely often. Let $s$ be a stage after which $R$ is never reinitialized. If $R$ is a $WP$, $D$ or $T$-requirement, then it can act only once more, after which it becomes inactive. Thus $R$ must be an $S$-requirement. For $R$ to injure $Q$, it does so in Cases 4.1, 4.2 or 4.3 infinitely often. It can do so in Cases 4.1 and 4.3 only once after stage $s$, as it is not reinitialized after stage $s$. Thus, it must infinitely often injure $Q$ via Case 4.2. In this case, it infinitely often increases the maximal $l$ for which $a^l$ is defined. We need only show that after infinitely many stages, this $l$ will permanently exceed $\#(Q)$. Otherwise, infinitely often, we must have a $T$-requirement with $\#(T) < \#(Q)$ which undefines $R$’s parameter $y^{\#(Q)}$. But each such $\#(T)$ can do this only once. This is because a $T_{j,k}$-requirement acts in Case 3.3 by making $k \in [W_s^i]_{E_{i,s}}$. But then it can never be in Case 3.3 again, as it would be in Case 3.1 instead. Thus, each $T$-requirement with $\#(T) < \#(Q)$ can undefine $R$’s parameter $y^{\#(Q)}$ at most once, so it will eventually be permanently defined and $R$ will not reinitialize $Q$ after that.

□

Definition 3.24. Let $Q$ be a $WP^i_j$-requirement. We say that $x$ is an i-active $Q$-number at stage $s$ if $x$ is a parameter of $Q$ at stage $s$. Let $Q$ be an active $Di^i_j$-requirement with parameters $x, x', z$. Then we say that $x, x'$ are i-active $Q$-numbers at stage $s$ and $z$ is an $i'$-active $Q$-number at stage $s$. Let $Q$ be an $S$-requirement and $x$ be a parameter of $Q$. We say that $x$ is a 0-active and $n$-active $Q$-number at stage $s$.

We say that $x$ is an $i + 1$-active $C_i$-number at stage $s$ if it is defined to be $a^{i+1}_x$ for some $x$ so that $x$ is the least member of $[x]_{E_{i+1}}$ at stage $s$. If $i + 1 = n$, we also say that $x$ is 0-active.

Lemma 3.25. Let $x$ and $y$ be distinct $i$-active numbers at stage $s$. Then $x E^i y$.

Proof. This is clearly true at stage 0. Suppose $s + 1$ is the least stage at which this lemma fails. Let $x, Q_1, y$, and $Q_2$ witness this. Let us consider the action at stage $s$ which brought about this situation. At stage $s$, we must have done more than just defining new parameters, because all new parameters are chosen to be fresh. In particular, if $z$ is fresh, then $[z]_{E_{i+1}} = \{z\}$, so it cannot contribute to violating our lemma.

There are two parts of the construction at stage $s$: The action in each of the 4 cases, and then the coding step. We verify that after each of these actions, we have not violated our lemma.

In each of cases 1, 2, and 4 where we can cause a collapse, we have a requirement $Q$ which $E_i$-collapses one of its parameters $z$ to some other element $w$. We then declare $Q$ to be inactive. By inductive hypothesis, $[z]_{E_i}$ contains only one $i$-active number, namely $z$. Thus, since $z$ is not $i$-active at stage $s + 1$ since $Q$ becomes inactive, we have added no new active numbers to $[w]_{E_i}$.

In case 3 where $Q = T_{j,k}^i$ where we cause collapse, we collapse $k$ (which may be equivalent to an active number for another requirement) with $x$, which is $> r$ and not equivalent to $a^l_j$ for any $l$. Since $x > r$, it is not equivalent to any parameter for a higher-priority $WP$ or $D$ requirement.
If it is equivalent to the parameter of a higher-priority $S$-requirement, then we undefine the $S$-requirement’s parameter. Similarly, if it is equivalent to a lower-priority requirement’s parameter, then we undefine this parameter via reinitialization. Either way, we add no new $i$-active numbers to the $E_i$-class of $k$.

Lastly, we have to check that our collapses in the coding step do not cause us to violate this lemma. These are of the form of $E_i$-collapsing $a_x$ with $a_y$ if we have $E_{i-1}$-collapsed $x$ with $y$. We can assume that prior to $E_{i-1}$-collapsing $x$ with $y$, both were least in their $E_{i-1}$-classes. Thus, both $a_x$ and $a_y$ were $i$-active. It follows that $[a_x]_{E_i}$ and $[a_y]_{E_i}$ had only one $i$-active element, namely $a_x$ and $a_y$. But since $x$ and $y$ have collapsed, one has stopped being active. So the newly formed class $[a_x]_{E_i} \cup [a_y]_{E_i}$ still contains only one $i$-active element.

Lemma 3.26. Let $x = a_k^{i+1}$. Then for every $s > k$ there exists a $j \leq k$ so that $xE_{i+1}^s a_j^{i+1}$ and $a_j^{i+1}$ is $i + 1$-active at stage $s$.

Proof. Let $j$ be the least number in $[k]_{E_i}$. Then $j \leq k$ and $a_j^{i+1}$ is $i + 1$-active. By the coding step, $jE_{i+1}$ implies $a_j^{i+1}E_{i+1}a_k^{i+1}$.

Lemma 3.27. Let $x$ be a number mentioned before stage $s$. Suppose that $x$ is not $E_{i+1}$-equivalent to any $i$-active number at stage $s$. Then for all $t > s$, $x$ is not $E_{i+1}$-equivalent to any $i$-active number at stage $t$.

Proof. Consider the first stage at which $x$ becomes $E_i$-equivalent to an $i$-active number. This cannot be caused by our assignment of parameters, since all parameters are assigned to be new. By the same analysis as in Lemma 3.25, any active $z$ which is collapsed with $x$ must simultaneously become inactive. Similarly, this cannot be caused by collapsing for the sake of coding, as this collapses two $E_i$-classes which already contain $i$-active numbers by Lemma 3.26.

Corollary 3.28. For every $i < n$ and $j, k < s$, $jE_{i+1}^sk$ if and only if $a_j^{i+1}E_{i+1}^sk$.

Proof. In the Coding stage, we guarantee that $jE_{i+1}^sk$ implies that $a_j^{i+1}E_{i+1}^sk$. To see the reverse, suppose that $jE_{i+1}^sk$ and let $j_0$ be least in $[j]_{E_i}$ and $k_0$ be least in $[k]_{E_i}$. It follows by construction that $a_j^{i+1}E_{i+1}^s a_j^{i+1}$ and $a_k^{i+1}E_{i+1}^s a_k^{i+1}$. Then $a_j^{i+1}$ and $a_k^{i+1}$ are both active numbers at stage $s$. It follows by Lemma 3.25 that they cannot be $E_{i+1}$-equivalent. Thus $a_j^{i+1}E_{i+1}^sk$.

Lemma 3.29. Suppose that $x$ and $y$ are numbers considered before stage $s$ and $x \not{E_s^k} y$. Suppose that $Q$ is a requirement which is deactivated at stage $s$ (thus all lower-priority requirements are reinitialized at stage $s$). Suppose further that $Q$ is never reinitialized after stage $s$. Then $x \not{E_s^k} y$.

Proof. Case 1: Neither $x$ nor $y$ are $E_s^k$-equivalent to any $i$-active number at stage $s$. Then by Lemma 3.27 this is true at every $t > s$. Then at any stage $t > s$ we cannot collapse $x$ with $y$ by a WP, D, or $S$-strategy or during the coding step because neither are equivalent to any active numbers. Lower priority $T$-requirements will have parameters $r > x, y$. Thus these cannot cause the collapse either. No higher priority $T$-requirement can cause the collapse as this would reinitialize $Q$.

\footnote{We define $0 - 1 = n - 1$.}
Case 2: Suppose that either \( x \) or \( y \) is \( E_t^i \)-equivalent to an \( i \)-active \( Q \)-number at stage \( s \) for \( Q \) a higher-priority \( WP \), \( D \) or \( S \)-requirement. Without loss of generality, we suppose this is true of \( x \). Suppose that \( xE_t^iy \). Let \( t > s \) be the stage at which we cause this collapse. Since \( Q \) does not act after stage \( s \), we know that \( x \) is also \( E_t^i \)-equivalent to an \( i \)-active \( Q \)-number at stage \( t \), and thus cannot be \( E_t^i \)-equivalent to any other \( i \)-active number by Lemma 3.25. Thus, the collapse must be caused by an active number \( E^i \)-equivalent to \( y \) or a \( T \)-requirement. It cannot be due to a \( T \)-requirement by the same reason as in case 1. So we now have three cases to consider:

Case 2a: Suppose that \( y \) is not \( E_t^i \)-equivalent to an \( i \)-active number at stage \( s \). Then by lemma 3.27, this is true at stage \( t \) also, so the collapse cannot occur at stage \( t \).

Case 2b: Suppose that \( y \) is \( E_t^i \)-equivalent to \( a_j^i \) for some \( j \). Then by Lemma 3.4, \( y \) is \( E_t^i \)-equivalent to some \( i \)-active \( a_k^i \) for some \( k \). Thus, it is not \( E_t^i \)-equivalent to a \( Q' \)-number at stage \( t \) for \( Q' \) any \( WP \), \( D \) or \( S \)-requirement. Since \( x \) cannot be equivalent to any \( a_j^i \) (since Lemma 3.26 shows that it would then be \( E_t^i \)-equivalent to two \( i \)-active numbers contradicting Lemma 3.25), the collapse can also not occur due to the coding.

Case 2c: Suppose that \( y \) is also \( E_t^i \)-equivalent to an \( i \)-active \( Q' \)-number at stage \( s \) for a higher-priority \( WP \), \( D \) or \( S \)-requirement. Then, \( y \) is also \( E_t^i \)-equivalent to a \( Q' \)-number at stage \( t \), and thus cannot be \( E_t^i \)-equivalent to any other active number by Lemma 3.25. Thus we cannot cause the collapse at stage \( t \).

Case 3: Suppose that \( xE_t^ia_j^i \) and \( y \) is not \( E_t^i \)-equivalent to any \( i \)-active number. Then Lemma 3.26 shows that \([x]_{E_t^i} \) always contains an \( i \)-active number for every \( t > s \) and Lemma 3.27 shows that \( y \) is never \( E_t^i \)-equivalent to an \( i \)-active number. Thus \( x \) and \( y \) can not be \( E_t^i \)-equivalent for any \( t > s \).

Case 4: Both \( xE_t^ia_j^i \) and \( yE_t^ia_k^i \). Then the only cause of the collapse of \( xE_t^iy \) is due to the coding step, since neither can ever be \( E_t^i \)-equivalent to any other \( i \)-active number and \( T \)-requirements only use numbers not equivalent to \( a_j^i \) for any \( l \). But then we can consider why we collapse \( jE_t^ia_{i-1}k \). By cases 1-3, the only possibility is that this, in turn was via a coding step. But the coding step at any given stage is finite and originates in a collapse for a \( WP \), \( D \), \( T \), or \( S \)-requirement, which we have ruled out in the cases above.

\[ \square \]

**Lemma 3.30.** Each requirement is satisfied.

**Proof.** Suppose towards a contradiction that \( Q \) is the highest priority requirement which is not satisfied. Let \( s \) be a stage after which \( Q \) will not be reinitialized. We consider the cases:

\( Q = WP_j^i \): Once the parameter \( x \) is chosen after stage \( s \), this is permanent. Either \( \varphi_j(x) \) diverges or we collapse \( xE_t^i\varphi_j(x) \). Either way, the requirement is satisfied.

\( Q = D_j^{ix} \): Once the parameters \( x, x', z \) are chosen after stage \( s \), this choice is permanent. If \( \varphi_j(x) \) or \( \varphi_j(x') \) do not converge or no \( \varphi_j(y) \) converges to equal \( z \), then \( \varphi_j \) is not a bijection and the requirement is satisfied. Otherwise let \( t \) be the first stage we consider \( Q \) after these convergences are witnessed. Similarly, if \( \varphi_j(x)E_t^i\varphi_j(x') \), then Lemma 3.29 guarantees that \( xE_t^i\varphi_j(x') \) and the requirement is satisfied. Otherwise, (possibly after reversing \( x \) and \( x' \)), we have \( \varphi_j(x) = w\varphi_t^jz \). Then we \( E_t^i \)-collapsed \( x \) with \( y \). Thus it suffices to show that \( w\varphi_t^jz \). This follows directly from Lemma 3.29.

\( Q = T_j^{i,k} \): Once the requirement chooses its parameter \( r \) after stage \( s \), this is permanent. Suppose \( W_j \) intersects infinitely many \( E_t \)-classes which do not contain any element of the form \( a_j^i \). Then \( W_j \)
will enumerate a number \(x\) which is not \(E_i\)-equivalent to any number \(a_i^j\) and also not \(E_i\)-equivalent to any number less than \(r\). Thus, once we consider the requirement after such a number \(x\) is enumerated into \(W_j\), we will either already have \(k \in [W_i^j]_{E_i}\) or we will collapse \(x\) with \(k\).

\[ Q = S_j: \text{We have two cases to consider:} \]

**Case A:** The sequence of parameters \(y^0, y^1, \ldots\) which are never removed is infinite. In this case, each of these \(y^i\) are \(i\)-active permanently. It follows that \(y^i \in E_0^{\pi y^i}\) for each pair \(l, l'\). For each one, we have \(\{[\varphi_j(y^0)]_{E_0}, [\varphi_j(y^i)]_{E_0}\} = \{[\pi^k y^0]_{E_0}, [\pi^k y^i]_{E_0}\} \) for some \(k\). We next check that this \(k\) is the same for each \(l\).

**Claim 3.31.** If \(k < k'\), then \(\pi^k(y^i) \in E_0^{\pi^k(y^i)}\) for any \(z\).

**Proof.** Suppose otherwise that \(\pi^k(y^0) \in E_0^{\pi^k(y^i)}\). Then since \(\pi\) is a reduction of \(E_0\) to itself, we get that \(y^0 \in E_0^{\pi^k(y^i)}\). But \(\pi^{k-k}(z)\) is a number of the form \(a_{w}^{n}\) for some \(w\). But then \(y^0\) is \(0\)-active and equivalent to another \(0\)-active number of the form of \(a_{w}^{n}\) by Lemma 3.26, which contradicts Lemma 3.25. \(\Box\)

It follows that for each \(l\) we have \(\varphi_j(y^i) \in E_0^{\pi^k(y^i)}\) for the same number \(k\). Note that we cannot have, for instance that \(\varphi_j(y^0) \in E_0^{\pi^k(y^i)}\) and \(\varphi_j(y^i) \in E_0^{\pi^k(y^0)}\), because then the condition \(\{[\varphi_j(y^0)]_{E_0}, [\varphi_j(y^i)]_{E_0}\} = \{[\pi^k y^0]_{E_0}, [\pi^k y^i]_{E_0}\}\) will fail for \(l = 2\). By the claim, we have that \(\varphi_j(y^i) \notin [\text{im}(\pi^{k+1})]_{E_0}\). Since \(y^i \in E_0^{\pi y^i}\) for each pair, we get that \(\pi^k(y^i) \in E_0^{\pi^k(y^i)}\), and we have that \(\text{im}(\varphi_j) \cap ([\text{im}(\pi^{k})]_{E_0} \setminus [\text{im}(\pi^{k+1})]_{E_0})\) contains infinitely many classes.

**Case B:** We only have finitely many stable parameters \(y_0, y_1, \ldots y^i\). By Lemma 3.22, there is a stage \(s'\) after which \(y^{i+1}\) is never defined. At a stage \(t > s'\), when the requirement is considered, it must either have \(\varphi_j(y^i)\) diverge, \(\varphi_j(y^0) \in E_0^{\pi y^i}\), in which case it does nothing, but \(y^0 \in E_0^{\pi y^i}\), since both remain \(0\)-active permanently, or it must have \(\varphi_j(y^0) \in E_0^{\pi y^i}\). In this latter case, we \(E_0\)-collapse \(y^0\) with \(y^i\). By Lemma 3.29, we see \(\varphi_j(y^0) \in E_0^{\pi y^i}\), so \(\varphi_j\) is not a reduction of \(E_0\) to itself, and the requirement is satisfied. \(\Box\)

**Lemma 3.32.** \(E_0, \ldots, E_{n-1}\) are all equivalent, yet non-isomorphic weakly precomplete ceers. They are all non-self-full.

**Proof.** By the requirements \(C_i\), we have \(E_0 \equiv_c E_1 \equiv_c \cdots \equiv_c E_n = E_0\), thus they are all equivalent, and by requirements \(D_{j}^{s'}\), they are non-isomorphic. By the requirements \(W^P_{j}\), they are all weakly precomplete. Since \(E_{n-1} \equiv_c E_n\) and \(E_{n-1} \not\equiv E_{n}\), we see by Lemma 2.3 that the reduction \(\pi_n\) is not onto the classes of \(E_n\). Therefore, the map \(\pi\) is not onto the classes of \(E_0\). But then \(\pi\) is a reduction of \(E_0\) to itself which is not onto the classes of \(E_0\), showing that \(E_0\) is non-self-full. Self-fullness is a property of degrees, so no \(E_i\) is self-full. \(\Box\)

**Lemma 3.33.** A ceer \(R\) is equivalent to \(E_0\) if and only if it is isomorphic to a ceer of the form \(E_i \oplus D\) where \(D\) is a ceer with either finite or infinite domain and is comprised of finitely many computable classes.

**Proof.** Suppose \(R\) is of the form \(E_i \oplus D\) where \(D\) is a ceer comprised of finitely many computable classes. Then \(R \equiv_c E_i \oplus \text{Id}_n\) where \(n\) is the number of classes in \(D\). But since \(E_i\) is non-self-full, we have that \(E_i \oplus \text{Id}_n \equiv_c E_i\).
Suppose $R \equiv_c E_0$. Then let $\varphi_j$ be the reduction given by $E_0 \leq_c R \leq_c E_0$, and let $\psi$ be the reduction of $R$ to $E_0$. Note that $\im(\varphi_j) \subseteq \im(\psi)$. By $S_j$, for some $k$, $\im(\varphi_j)]_{E_0} \cap [\im(\pi^k)]_{E_0} \setminus [\im(\pi^{k+1})]_{E_0}$ is infinite, and, therefore, $K = [\im(\psi)]_{E_0} \cap [\im(\pi^k)]_{E_0} \setminus [\im(\pi^{k+1})]_{E_0}$ is infinite too. Let $k$ be least so that $[\im(\psi)]_{E_0} \cap [\im(\pi^k)]_{E_0} \setminus [\im(\pi^{k+1})]_{E_0}$ is infinite. Then $\im(\psi)$ intersects only finitely many $E_0$-classes from $\omega \setminus [\im(\pi^k)]_{E_0}$ with witnesses, say, $c_1, c_2, \ldots, c_n$. Since $K$ is the union of the sets $K_m = [\im(\psi)]_{E_0} \cap [\im(\pi^k \circ \pi_n \circ \cdots \circ \pi_{m+1})]_{E_0} \setminus [\im(\pi^k \circ \pi_n \circ \cdots \circ \pi_m)]_{E_0}$, one of them is infinite. Let $m$ be the biggest such number. Then $\im(\psi)$ intersects only finitely many $E_0$-classes from $[\im(\pi^k)]_{E_0} \setminus [\im(\pi^k \circ \pi_n \circ \cdots \circ \pi_{m+1})]_{E_0}$. Let $d_1, d_2, \ldots, d_{n_2}$ be witnesses of these classes.

Now, consider the c.e. set $W = \{ i \mid \pi^k \circ \pi_n \circ \cdots \circ \pi_{m+1}(i) \in [\im(\psi)]_{E_0}\}$. Then this $W$ hits infinitely many $E_m$-classes which are not in the range of $\pi_m$. By the $T$-requirements, it intersects every $E_m$-class. Thus, $\im(\psi)$ contains $\im(\pi^k \circ \pi_n \circ \cdots \circ \pi_{m+1})$ along with finitely many more classes $[c_j]_{E_0}, [d_k]_{E_0}$. Hence, in $R$, the set of $i$ so that $\psi(i) \in [\im(\pi^k \circ \pi_n \circ \cdots \circ \pi_{m+1})]$ is c.e. and the set of $i$ so that $\psi(i)$ is in $\bigcup[c_j]_{E_0} \cup [d_k]_{E_0}$ is c.e., and this gives a finite partition of $\omega$ into c.e. sets. Therefore, these sets are computable. Thus $R$ is equivalent to the uniform join of $E_0$ restricted to the set $\im(\pi^k \circ \pi_n \circ \cdots \circ \pi_{m+1})$ and a c.e. with finitely many computable classes. But $E_0$ restricted to the set $\im(\pi^k \circ \pi_n \circ \cdots \circ \pi_{m+1})$ is isomorphic to the c.e $E_m$ by Lemma 2.3.

Corollary 3.34. There is a weakly precomplete c.eer $E$ so that for any c.eer $R$, $R \equiv_c E$ if and only if $R$ is isomorphic to $E \oplus X$ where $X$ is a c.e. equivalence relation (on a possibly finite universe) comprised of finitely many computable classes.

Proof. Apply the previous construction with $n = 1$. Note that we have no $D$-requirements, and thus we have to work slightly harder to ensure non-self-fullness. It suffices to ensure that the class of 0 is not equivalent to any $a_i$, and to do this it suffices by Lemma 3.27 to begin the construction by mentioning the number 0.

4. Ceers reducible to one with finite classes

Theorem 4.1. The index set of the collection of ceers reducible to one with only finite classes is $\Sigma^0_4$-complete.

Proof. This proof is a standard priority construction using a tree of strategies. This is somewhat unusual in the study of ceers, where most arguments are finite injury arguments.

It is easy to estimate that the desired index set is $\Sigma^0_4$. To prove the theorem we fix a $\Sigma^0_4$-complete set $S = \{ i \mid \exists j W_g(i,j) \text{ is co-infinite} \}$ and consider requirements:

$$P_{j,k} : \text{If } [k, \omega] \subseteq \bigcap_{m \leq j} W_g(i,m), \text{ then } \varphi_j \text{ is not a reduction of } E \text{ to a c.eer with only finite classes.}$$

$P_{j,k}$-strategy: Step 1: Let $x_0$ be a fresh number.

Step 2: Wait for $\varphi_j(x_i)$ to converge for every $x_i$ which has been chosen. If $\varphi_j$ is injective on the set of chosen $x_i$, then go to step 3. Otherwise, we will have $x_1, x_2$ so that we keep $x_1, E x_2$, yet $\varphi_j(x_1) = \varphi_j(x_2)$ showing that $\varphi_j$ is not a reduction of $E$ to any c.eer.
Step 3: Collapse each defined \( x_l \) to be \( E \)-equivalent to \( x_0 \). Let \( n \) be least so \( x_n \) is not yet defined. Choose \( x_n \) to be fresh and go back to Step 2.

We put these strategies on a tree as usual for an infinite-injury construction. We will omit some details of the construction in favor of clarity. We fix a tree of strategies with nodes \( \{ \infty, f \}^{< \omega} \). Each strategy on level \( \langle j, k \rangle \) of the tree will be a \( P_{j,k} \)-strategy. If a node \( \beta \) is a \( P_{j,k} \)-strategy, and \( \beta \geq \alpha \infty \) where \( \alpha \) is a \( P_{j,k} \)-strategy, then we say \( \beta \) is redundant and it never acts.

When visited, a \( P_{j,k} \)-strategy will have outcomes \( \infty < f \). A stage is expansionary for \( \beta \), a \( P_{j,k} \)-node if the least element of \( [k, \omega) \setminus \bigcap_{m \in j} W_{g(i,m)} \) is larger at the current stage then at the last stage when \( \beta \) was visited. \( \beta \) only acts on stages where \( \beta \) is visited which are expansionary for \( \beta \). If it acts and it goes to step 3 (thus choosing a new \( x_n \)), it will take the outcome \( \infty \). Otherwise, it takes the outcome \( f \). As usual, we define the current path by the outcomes taken by nodes visited, and if we visit a node left of \( \beta \), then we re-initialize \( \beta \). This concludes the description of the construction.

Note that since the strategies each work with fresh elements \( x_i \) and only collapse them to other elements chosen by that strategy, if \( \alpha \) chooses \( x_l \) and \( x_n \) and does not choose to collapse \( x_l \) with \( x_n \), then \( x_l E x_n \).

Lemma 4.2. Suppose that for every \( j \), \( W_{g(i,j)} \) is co-finite. Then no \( \varphi_j \) is a reduction of \( E \) to a ceer whose classes are finite.

Proof. Fix \( j \) and let \( k \) be least so that \( [k, \omega) \subseteq \bigcap_{m \in j} W_{g(i,m)} \). Let \( t_p \) be the true path, let \( \alpha \) be the \( P_{j,k} \)-strategy on \( t_p \). Let \( s \) be a stage large enough that no node left of \( \alpha \) is visited after stage \( s \). Thus, at any \( \alpha \)-expansionary stage \( t > s \), we define \( x_0 \) to have its final value. We consider two cases: \( \alpha \infty \leq t_p \) or \( \alpha f \leq t_p \). In the first case, \( \varphi_j \) is injective on the set \( \{ x_l \mid l \in \omega \} \), but we make each of these \( E \)-equivalent. Suppose \( E \leq_c R \) is witnessed by the reduction \( \varphi_j \). Then \( \{ \varphi_j(x_m) \mid m \in \omega \} \) either defines more than one \( R \)-class, in which case \( \varphi_j \) is not a reduction of \( E \) to \( R \), or it defines an infinite subset of one class, showing that \( R \) has an infinite class.

In the second case, the strategy gets stuck in step 2: This means that either \( \varphi_j \) is not total or for some \( l < k \), we have \( \varphi_j(x_l) = \varphi_j(x_k) \), but we never collapse \( x_l \) and \( x_k \). Thus, \( \varphi_j \) is not a reduction of \( E \) to any ceer. \( \square \)

Lemma 4.3. Suppose that for some \( j \), \( W_{g(i,j)} \) is co-infinite. Then \( E \) is reducible to a ceer with finite classes.

Proof. Let \( j \) be least so that \( W_{g(i,j)} \) is co-infinite. Then for every \( j' \geq j \) and any \( k' \), any \( P_{j',k} \)-strategy has only finitely many expansionary stages. Thus, each strategy can only create finite classes. Thus, the only strategies which can create infinite classes in \( E \) are the strategies on the true path which are \( P_{j',k} \)-strategies with \( j' < j \). But there are only finitely many of these which are not redundant – at most one for each \( j' < j \). Thus \( E \) is a ceer with at most \( j \) infinite classes. But note that each of these classes are computable: The strategy chooses \( x_0 < x_1 < x_2 < \ldots \) and this forms the class. We can let \( R \) be the ceer formed by replacing each of these classes by a single point. It is easy to see that \( E \leq_c R \), and \( R \) has only finite classes. \( \square \)

Gao and Gerdes [12] gave an indirect proof that there is a ceer \( E \) all of whose classes are computable, but \( E \) is not reducible to any ceer with only finite classes. They do this by showing that the index set of ceers with all computable classes is \( \Pi^0_3 \)-complete, but the index set of ceers reducible to one
with only finite classes is $\Sigma^0_4$ (they only show $\Pi^0_3$-hardness). They ask for a direct construction of such a ceer.

**Observation 4.4.** The proof of Theorem 4.1 gives a direct construction of a ceer with computable classes which does not reduce to one with only finite classes.

**Proof.** For any $i \notin S$, the ceer $E$ produced in the previous construction has computable classes (again, since the classes are chosen as $x_0 < x_1 < x_2 < \ldots$), but does not reduce to a ceer with only finite classes. □

5. Strong minimal covers of sets of degrees of ceers

We now turn our attention to some questions about least or minimal upper bounds for some subsets in the structure of ceers. Gao and Gerdes [12] asked whether $\text{Id}^n_1$ is a least upper bound of $\langle \text{Id}^n_1, k \rangle_{k \in \omega}$ and Andrews and Sorbi [4] asked whether there is a minimal upper bound for the set $\{\text{Id}^{(n)}_n \mid n \in \omega\}$.

**Definition 5.1.** If $S$ is a subset of a preordered set $\langle P, \leq \rangle$, we say that $c \in P$ is a strong minimal cover of $S$ if $c \not\in S$ and for every $x \in P$, $x \leq c \iff$ either $x = c$ or $\exists y \in S (x \leq y)$.

As usual, we write shortly $x \equiv y$ if $x \leq y \& y \leq x$ and write $x < y$ if $x \leq y \& y \not\leq x$.

Obviously, every strong minimal cover of $S$ is an upper bound for $S$. We will deal with *internally unbounded* subsets of a preordered set $P$, i.e. subsets $S$ that have a following property: $\forall x \in S \exists y \in S(x < y)$. For instance, the sets $\{\text{Id}_n \mid n \in \omega\}$ and $\{\text{Id}^{(n)}_n \mid n \in \omega\}$ are internally unbounded.

**Lemma 5.2.** (i) Let $S$ be a subset of a preordered set $\langle P, \leq \rangle$ that has a least upper bound $b$ and a strong minimal cover $c$. If $b \in S$ then $b < c$, otherwise, $c \equiv b$.

(ii) If $S$ is internally unbounded set then

- a strong minimal cover of $S$ is a minimal upper bound of $S$;
- if $S$ has two incomparable strong minimal covers then $S$ has no least upper bound.

Proof is obvious.

**Theorem 5.3.** Let $(E_i)$ be a uniform c.e. sequence of non-universal ceers. Then $\{\oplus_{i \in \omega} E_i \mid i \in \omega\}$ has infinitely many incomparable strong minimal covers.

**Proof.** We build infinitely many ceers $R_k$. Throughout the construction, we will have some columns of $R_k$ reserved for coding. If we reserve the $j$th column of $R_k$ as a coding column for $E_i$, then for every $x, y$, we ensure that $\langle j, x \rangle R_k \langle j, y \rangle$ if and only if $x \equiv y$. We say that a column is destroyed if every number in the column is equivalent to a number in a smaller column. We will ensure that if $x, y$ are in different coding columns, then $x \not\equiv y$.

We construct $(R_k)_{k \in \omega}$ to satisfy the following requirements:

- $C_{n,k}$: There is an $R_k$-coding column for $E_n$, i.e., there is a $j$ so that $\forall x, y(\langle j, x \rangle R_k \langle j, y \rangle \Leftrightarrow x \equiv y)$
- $P_{i,j,k}$: If $W_i$ intersects the closures of infinitely many non-destroyed columns in $R_k$, then $W_i$ intersects $[j]_{R_k}$.
- $D_{c,k,k'}$: The function $\varphi_c$ does not give a reduction of $R_k$ to $R_{k'}$ if $k \neq k'$. 


For convenience, we choose to build each \( R_k \) so that the first column of \( R_k \) is exactly \( E_0 \) and the second column is exactly \( E_1 \). These two columns are coding columns, and this is not subject to injury.

\( C_n \)-strategies: Pick a new column and decide to code \( E_n \) into this column. Restrained this column from being destroyed by a lower priority \( R_{i,j} \)-requirement.

\( P_{i,j} \)-strategies: Wait for some \( x \) to enter \( W_i \) which is in a column \( > j \) which is not restrained by any higher-priority strategy from being destroyed. At this point, collapse the entire column of \( x \) to be equivalent to \( j \). We say this column has been destroyed.

\( D_{e,k,k'} \)-strategies: We first pick a new column \( j \) of \( R_k \). As long as it appears that \( \varphi_e \) gives a reduction of \( R_k \) into restrained columns of \( R_{k'} \), we will threaten to code a universal ceer on this column of \( R_k \). We will argue below that we do not succeed in this coding (in brief, this is because the restrained \( R_{k'} \)-columns will together be equivalent to a finite uniform join of the non-universal \( E_i \)'s, but the universal degree is uniform-join irreducible \([5]\), but the threat will suffice to guarantee that \( \varphi_e \) is not a reduction of \( R_k \) into the restrained columns of \( R_{k'} \). If the image of \( \varphi_e \) contains two classes in non-restrained columns, then we will explicitly diagonalize. We fix \( T \) a universal ceer.

Step 1: We use a parameter \( n \) which begins with \( n = 1 \). We choose \( a_0 = \langle j, 0 \rangle \) and \( a_1 = \langle j, 1 \rangle \). If we see a stage \( s \) so that \( \{ a_i \mid i \leq n \} \subseteq \text{domain}(\varphi_e) \) and for each \( x, y \leq n \), \( a_x R_k a_y \leftrightarrow \varphi_e(a_x) R_{k'} \varphi_e(a_y) \), then we \( R_k \)-collapse each pair \( a_x, a_y \) with \( x, y \leq n \) so that \( xT^s y \). We choose the least element of the \( j \)-th column which is not \( R_k^e \) equivalent to any \( a_i \) with \( i \leq n \), and let this be \( a_{n+1} \), we increment \( n = n + 1 \). While we wait for these convergences and equivalences, at stage \( s \), we collapse any element of \( \{ \langle j, i \rangle \mid i \leq s \} \sim \{ a_i \mid i \leq n \} \) with \( a_0 \) (while doing this, we do not say \( D \) is acting, and we do not reinitialize lower priority requirements – we do that when we increment \( n \) or act as in step 2). If, for some \( x, y \), \( a_x R_k^e a_y \), \( \varphi_e(a_x) \) and \( \varphi_e(a_y) \) converge and are not in columns of \( R_{k'} \) which are restrained by higher priority requirements, then we go to step 2.

Step 2: If \( \varphi_e(a_x) R_{k'}^{e} \varphi_e(a_y) \), then we destroy the \( j \)-th column of \( R_k \) by making every element of \([a_x]_{R_k^e} \) equivalent to \( \langle 0, 0 \rangle \) and every other element equivalent to \( \langle 1, 0 \rangle \).

Otherwise, we destroy the \( j \)-th column of \( R_k \) by making every element equivalent to \( \langle 0, 0 \rangle \). In addition, we will destroy the columns of \( \varphi_e(a_x) \) and \( \varphi_e(a_y) \) in \( R_{k'} \) as follows: If \( \varphi_e(a_x) \) and \( \varphi_e(a_y) \) are in different columns of \( R_{k'} \), then destroy the column of \( \varphi_e(a_x) \) by making every element equivalent to \( \langle 0, 0 \rangle \) and destroy the column of \( \varphi_e(a_y) \) by making every element equivalent to \( \langle 1, 0 \rangle \). Lastly, if \( \varphi_e(a_x) \) and \( \varphi_e(a_y) \) are in the same column of \( R_{k'} \), then we destroy the column by making every element in \(([\varphi_e(a_x)]_{R_{k'}} \) equivalent to \( \langle 0, 0 \rangle \) and every other number in the column equivalent to \( \langle 1, 0 \rangle \).

Construction. We construct a supplementary partial computable function \( d(e, k, k', s) \) along with building the ceers \( R_k \). We fix an ordering of all requirements in order type \( \omega \) and fix a computable correspondence between the stages of the construction and the requirements so that: at any stage \( s + 1 \) of the construction, we deal exactly with one of the requirements; every requirement is considered at infinitely many stages.

We fix a universal ceer \( T \) and its approximation \( T^s \). Let \( E_n^s \) be a computable approximation of a uniformly c.e. sequence \( E_n \) of non-universal ceers.
Reinitializing a $C_{n,k}$-strategy means that a coding column of equivalence $E_n$ in $R_k$, if any, is stopped being a coding column. Reinitializing a $D_{e,k,k'}$-strategy at stage $s+1$ means to set $d(e,k,k',s+1)$ undefined. Reinitializing a $P_{i,j,k}$-strategy means doing nothing.

Stage 0. Initialize all requirements. For every $k$, set $R_k^0 = Id$.

Stage $s + 1$ for $C_{n,k}$-requirement with $n > 1$. If $R_k$ has no chosen coding column for $E_n$, then pick a new, say $j$th, column in $R_k$ with $j > 1$ and declare this column to be the chosen coding column of $E_n$ in $R_k$. Restraine the $j$th column of $R_k$ from being destroyed by a lower priority requirement.

Correct all coding: for every $n', k'$, if the $j'$-th column of $R_{k'}$ is a coding column of $E_{n'}$, and $x E_{n'}^{s+1} y$, then $R_{k'}$-collapse $\langle j', x \rangle$ with $\langle j', y \rangle$.

Stage $s + 1$ for $P_{i,j,k}$-requirement. If the equivalence class $[j]_R^k$ does not intersect with and there are $x$ and $j' > 1$ so that $\langle j', x \rangle \in [W_i^k]_R^k$, $j < j'$, and the $j'$-th column in $R_k$ is not yet destroyed and not restrained by a strategy of higher priority from being destroyed, then $R_k$-collapse $j$ with the entire $j'$-th column. This means that we add $\langle j, \langle j', z \rangle \rangle$ into the computable equivalence relation $R_k^{s+1}$ for every $z$. We say that the $j'$-th column in $R_k$ has been destroyed. Reinitialize all strategies of lower priority.

Stage $s + 1$ for $D_{e,k,k'}$-requirement. We distinguish the following three cases.

Case 1. If $d(e,k,k',s)$ is not defined, then pick a new column $j > 1$ in $R_k$ and define $d(e,k,k',s + 1) = \langle j, 1 \rangle$, denote $\langle j, 0 \rangle$ and $\langle j, 1 \rangle$ by $a_0$ and $a_1$. Restraine the $j$th column of $R_k$ from being destroyed by a lower priority requirement. Reinitialize all strategies of lower priority.

Case 2. If $d(e,k,k',s)$ is defined and $\langle j, n \rangle$, $n \geq 1$, $\{a_i \mid i \leq n\} \subseteq \text{domain}(\varphi_e^a)$ and, for each $x, y \leq n$, $a_x R_k^{s} a_y \Leftrightarrow \varphi_e(a_x) R_{k'}^{s} \varphi_e(a_y)$, then check whether there exist $x, y$ so that:

1. $0 \leq x < y \leq n$,
2. $a_x R_k^{s} a_y$,
3. $\langle \varphi_e(a_x) \rangle_0 > 1$ and $\langle \varphi_e(a_y) \rangle_0 > 1$,
4. $\varphi_e(a_x)$ and $\varphi_e(a_y)$ are not in columns of $R_{k'}$ which are restrained by higher priority requirements from being destroyed.

If so, go to Subcase 2.1, otherwise, go to Subcase 2.2.

Subcase 2.1.

(i) If $\varphi_e(a_x) R_{k'}^{s} \varphi_e(a_y)$, then destroy the $j$th column of $R_k$ by $R_k$-collapsing every element of $[a_x]_R^k$ with $\langle 0, 0 \rangle$ and every other element of the $j$th column with $\langle 1, 0 \rangle$. Define $d(e,k,k',s+1) = \langle 0, 0 \rangle$. Reinitialize all strategies of lower priority.

(ii) If $\varphi_e(a_x) R_{k'}^{s} \varphi_e(a_y)$ and $\varphi_e(a_x)$ and $\varphi_e(a_y)$ are in different columns of $R_{k'}$, then: destroy the $j$th column of $R_k$ by $R_k$-collapsing every element of this column with $\langle 0, 0 \rangle$; destroy the column of $\varphi_e(a_x)$ in $R_{k'}$ by $R_{k'}$-collapsing every element of the column with $\langle 0, 0 \rangle$; and destroy the column of $\varphi_e(a_y)$ by $R_{k'}$-collapsing every element of the column with $\langle 1, 0 \rangle$. Define $d(e,k,k',s+1) = \langle 0, 0 \rangle$. Reinitialize all strategies of lower priority.

(iii) If $\varphi_e(a_x) R_{k'}^{s} \varphi_e(a_y)$ and $\varphi_e(a_x)$ and $\varphi_e(a_y)$ are in the same column of $R_{k'}$, then: destroy the $j$th column of $R_k$ by $R_k$-collapsing every element of this column with $\langle 0, 0 \rangle$; destroy the column of $\varphi_e(a_x)$ in $R_{k'}$ by $R_{k'}$-collapsing every element of $[\varphi_e(a_x)]_R^{k'}$ with $\langle 0, 0 \rangle$ and every other element in the column with $\langle 1, 0 \rangle$. Define $d(e,k,k',s+1) = \langle 0, 0 \rangle$. Reinitialize all strategies of lower priority.
Subcase 2.2.

(i) $R_k$-collapse each pair $a_x, a_y$ with $x, y \leq n$ so that $x T^* y$;
(ii) choose the least $\langle j, a \rangle$ which is not yet $R_k$-equivalent to any $a_i$ with $i \leq n$, define $d(e, k, k', s + 1) = a_{n+1} = \langle j, a \rangle$.

Case 3. If Cases 1, 2 do not hold, then $R_k$-collapse any element of $\{ \langle j, i \rangle \mid i \leq s \} \setminus \{ a_i \mid i \leq n \}$ with $a_0$.

End of stage $s + 1$. For each $k$ perform the symmetrical and transitive closure of the set of pairs that have been enumerated into $R_k$ by the end stage $s + 1$. Go to the next stage.

Verification. The verification is done via the following lemmas. We say that a column is an active column at stage $s$ if it has not been destroyed by stage $s$. Obviously, columns 0 and 1 are active at any stage.

Lemma 5.4. There is no stage $s$ and numbers $x$ and $y$ in different active columns at stage $s$ so that $x R^*_s y$.

Proof. We collapse elements within the same column at stages for $C$-requirements to correct coding or in Subcase 2.2 or 3 at stages for $D$-requirements followed by performing End of stage. These collapses are made without destroying columns. During other collapsing pairs of numbers of different columns, at least one of these columns is destroyed, thus no longer is an active column.

We say that a column of $R_k$ is a permanent coding column if it is chosen to code some $E_n$ into $R_k$ by a $C_{n,k}$ strategy which is never reinitialized after this choice.

Lemma 5.5. If $x$ and $y$ are in different permanent coding columns of $R_k$, then $x R_k y$.

Proof. From some stage onwards, both $x$ and $y$ are in active columns at stage $s$. Thus by the previous lemma, $x R^*_k y$ for each $s$, showing that $x R_k y$. □

Lemma 5.6. At any stage $s$, if $x$ is in the $j$th column, and the $j$th column has been destroyed, then $x$ is equivalent to an element in an active column.

Proof. When we destroy a column, we make every element equivalent to an element in a smaller column. Either this column is active, or it is destroyed, making every element equivalent to an element in a smaller column. Since $\omega$ is well-ordered, and 0 and 1 are coding columns in each $R_k$, we see that $x$ is equivalent to an element in an active column. □

Lemma 5.7. Each strategy reinitializes lower priority strategies only finitely often. For every $e, k, k'$, $\lim_{s \to \omega} d(e, k, k', s)$ is finite. And each requirement is satisfied.

Proof. We show the result by induction on the priority of requirements, identifying the claim that $\lim_{s \to \omega} d(e, k, k', s)$ is finite with the requirement $D_{e,k,k'}$. So, we may assume that every strategy of higher priority than $\mathcal{S}$ reinitializes lower priority requirements only finitely often, is satisfied, and all higher priority $D_{e,k,k'}$-requirements have $\lim_{s \to \omega} d(e, k, k', s)$ finite.

Let $s_0$ be the least stage of the construction so that after stage $s_0$ each strategy of higher priority than $\mathcal{S}$ does not reinitialize lower priority strategies after stage $s_0$. If $\mathcal{S}$ is a $C_{n,k}$-strategy, then it never reinitializes lower-priority strategies and once it chooses a coding column after stage $s_0$,
this choice of column is permanent, and it succeeds in coding $E_n$ into this column of $R_k$. If $S$ is $P_{i,j,k}$-strategy, note that it can reinitialize lower priority strategies at most once after stage $s_0$. Note that since each higher priority strategy only restrains at most one column of $R_k$, if $W_i$ intersects the closures of infinitely many non-destroyed columns of $R_k$, then it will intersect the closure of one which is not restrained by a higher-priority requirement, and $S$ will act satisfying the requirement.

Lastly, we consider the case that $S$ is a $D_{e,k,k'}$-strategy. A value of the function $\lambda s d(e, k, k', s)$ may be undefined or be any natural number. Note that $D_{e,k,k'}$-strategy reinitializes lower-priority strategies only by Subcase 2.1 and if it did so at stage $s_1 + 1 > s_0$ then $d(e, k, k', s_1)$ equals to some number $\langle j, n \rangle$ with $n \geq 1$ while $d(e, k, k', s_1 + 1) = \langle 0, 0 \rangle$. Besides, $d(e, k, k', s) = \langle 0, 0 \rangle$ for all $s \geq s_1 + 1$. Therefore, $D_{e,k,k'}$-strategy does not reinitialize lower-priority strategies after stage $s_1 + 1$, since only Case 3 holds at these stages. Note that if we enter Subcase 2.1, we explicitly diagonalize to ensure that $S$ is satisfied. This only uses that $\langle 0, 0 \rangle W_k(1, 0)$ for each $k$. Thus, we have the result if the strategy ever enters Subcase 2.1 after stage $s_0$.

Now, we prove that the $D_{e,k,k'}$-requirement is satisfied and $\lim_{s \to \infty} d(e, k, k', s)$ is finite, given that it is not reinitialized after stage $s_0$ and that after stage $s_0$, $S$ never enters Subcase 2.1. Since $d(e, k, k', s)$ can’t be undefined in all stages after the stage $s_0$, we can assume that $d(e, k, k', s_0)$ is defined due to Case 1. So, $d(e, k, k', s)$ is defined and is different from $\langle 0, 0 \rangle$ and Subcase 2.1 does not hold for all $s \geq s_0$. Suppose towards a contradiction that $\lim_{s \to \infty} d(e, k, k', s)$ is infinite. Then Subcase 2.2 holds infinitely often and $\varphi_e$ reduces a universal ceer $T$ to the closures of finitely many columns of $R_k$. Each of these are restrained by being destroyed by a higher-priority requirement. Since every higher-priority $D_{e',l,l'}$-requirement has $\lim_{s \to \infty} d(e', l, l', s)$ finite, its column is either destroyed or contains only finitely many classes via co-finitely being in Case 3. Thus, each of these columns in $R_k$ is either destroyed, or is a permanent coding column, or has only finitely many classes. Thus $T \leq_{c} E_i \oplus \text{Id}_m$ for some $m$. But since the universal degree is uniform-join irreducible [5], we must have that $T \leq_{c} E_i$ for some $i$, but this contradicts each $E_i$ being non-universal.

Thus we know that $\lim_{s \to \infty} d(e, k, k', s)$ is finite. Then we must always take Case 3 after some $s_1 > s_0$. Therefore, either $\varphi_e$ is not a total function or the equivalence $a_x R_k a_y \leftrightarrow \varphi_e(a_x) R_k' \varphi_e(a_y)$ fails for some $x, y \leq \lim_{s \to \infty} \langle d(e, k, k', s) \rangle 1$, and the strategy succeeds. Note that the $R_k$-closure of $\langle d(e, k, k', s) \rangle 0$ column consists of finitely many equivalence classes.

 Lemma 5.8. Each $R_k$ is a strong minimal cover for $\{ \oplus_{i \leq j} E_i \mid j \in \omega \}$.

Proof. Since each $E_i$ is coded into some column of $R_k$ and these columns have disjoint $R_k$-classes, we see that every ceer in $\{ \oplus_{i \leq j} E_i \mid j \in \omega \}$ is reducible to $R_k$. Now, suppose $X \leq_{c} R_k$ via a computable function $f$. Let $W_i$ be the image of $f$. If it intersects the closures of only finitely many non-destroyed columns, then we can reduce $X$ to the uniform join of the finitely many $E_i$ or finite ceers coded on these columns. Thus $X \leq_{c} \oplus_{i \leq m} E_i \oplus \text{Id}_n$ for some $m, n$. But then $X \leq_{c} \oplus_{i \leq m+n} E_i$. Otherwise, $W_i$ intersects the closures of infinitely many non-destroyed columns. Then, by the $P_{i,j}$-requirements, $W_i$ intersects every class. Since the reduction $X \leq_{c} R_k$ is onto the classes of $R_k$, we have that $X \equiv R_k$.

\hfill $\Box$
We apply Theorem 5.3 combined with Lemma 5.2 to get several corollaries below. To prove them we need to show that suitable sets of ceers are of the form covered by Theorem 5.3 and are internally unbounded.

Note that our first corollary provides another proof that there are infinitely many incomparable dark minimal ceers as in [5, Theorem 3.3].

**Corollary 5.9.** There are infinitely many strong minimal covers for \( \{ \text{Id}_n \mid n \in \omega \} \) and no least upper bound.

**Proof.** We show that the \( \leq_c \)-downward closure of \( \{ \text{Id}_n \mid n \in \omega \} \) is the same as the \( \leq_c \)-downward closure of \( \{ \oplus_{i \leq n} \text{Id}_i \mid n \in \omega \} \). The former is clearly internally unbounded, so we can apply Theorem 5.3 and Lemma 5.2 to yield the result.

To see that these two downward closures are equal, it suffices to see that the former is closed under uniform-join, which follows from \( \text{Id}_n \oplus \text{Id}_m \equiv_c \text{Id}_{n+m} \).

**Corollary 5.10.** There are infinitely many minimal upper bounds for \( \{ \text{Id}_n' \mid n \in \omega \} \) and no least upper bound.

**Proof.** Similarly, we need only show that the downward-closure of this collection is closed under uniform join. It is not difficult to see that \( \text{Id}_n' \oplus \text{Id}_m' \leq \text{Id}_{n+m}' \). In fact, \( E' \oplus R' \leq (E \oplus R)' \) holds for all ceers by [4, Lemma 2.3].

**Observation 5.11.** In fact, \( \text{Id}' \) is also not a minimal upper bound of \( \{ \text{Id}_n' \mid n \in \omega \} \).

**Proof.** Consider the ceer \( \oplus_{n \in \omega} \text{Id}_n' \). This is an upper bound of \( \{ \text{Id}_n' \mid n \in \omega \} \), and a direct reduction shows that it reduces to \( \text{Id}' \). But since jumps are uniform join irreducible [4], this is strictly below \( \text{Id}' \).

The following answers a question from [4]:

**Corollary 5.12.** There are infinitely many strong minimal covers and no least upper bound to the set \( \{ \text{Id}^{(n)}_n \mid n \in \omega \} \).

**Proof.** Again, it suffices to show that for \( n \leq m \), \( \text{Id}^{(n)}_n \oplus \text{Id}^{(m)}_n \leq_c \text{Id}^{(m)}_m \). This is true due to [4, Lemma 2.3] that states: \( R' \oplus S' \leq_c (R \oplus S)' \) for any ceers \( R, S \). Then by induction, we get: \( R^{(n)} \oplus S^{(n)} \leq_c (R \oplus S)^{(n)} \) for any \( n \). Therefore
\[
\text{Id}^{(n)}_n \oplus \text{Id}^{(m)}_n \leq_c \text{Id}^{(m)}_m \oplus \text{Id}^{(m)}_m \leq_c (\text{Id} \oplus \text{Id})^{(m)}_m = \text{Id}^{(m)}_m.
\]

6. Observations on minimal ceers.

Gao and Gerdes [12] showed that: \( \text{Id}_1 \leq_c \text{Id}_2 \leq_c \text{Id}_3 \leq_c \cdots \leq_c \text{Id} \) and for every \( n > 1 \) and any ceer \( R \) with infinitely many classes, \( \text{Id}_n \leq_c R \). Further, every ceer with finitely many classes is \( \equiv_c \) to one of \( \text{Id}_n \). This implies that, when we examine the notion of a minimal ceer, we should consider minimality within the collection of ceers with infinitely many classes.

**Definition 6.1.** We call a ceer \( R \) with infinitely many classes to be minimal if, for every ceer \( S \), if \( S \leq_c R \) and \( S \) has infinitely many classes then \( S \equiv_c R \).
This immediately implies that $\text{Id}$ is the natural example of a minimal ceer. The following minimality criterion was used by Andrews and Sorbi \cite{AndrewsSorbi} to construct minimal ceers, but it was not known to be an equivalence. We here show the other implication.

**Theorem 6.2.** A ceer $R$ with infinitely many classes is minimal if and only if $R \equiv_c \text{Id}$ or, for every c.e. set $W$, if $W$ hits infinitely many $R$-classes then it hits every $R$-class.

**Proof.** Let $R$ be any minimal ceer and let a c.e. set $W$ hit infinitely many but not all $R$-classes. Let us show that $R \equiv_c \text{Id}$. Pick a number $a$ so that $[a]_R \cap W = \emptyset$. We choose a computable function $f$ with range $W$ and define a seer $S$ by $xSy \iff f(x)Rf(y)$. Then $S \leq_c R$ via the function $f$. Since $R$ is minimal and $S$ has infinitely many classes it follows that $xSy \iff g(x)Sg(y)$ for some computable function $g$ and all $x, y$. Note that $a \in \omega \setminus [\text{im}(f \circ g)]_R$, and, therefore, $aR(f \circ g)(a)$. This immediately implies that $a, (f \circ g)(a), (f \circ g)(2)(a)$ are pairwise non-equivalent relative to $R$. By iterating the function $f \circ g$ on $a$, we obtain an infinite c.e. sequence of numbers lying in distinct $R$-classes. If $h$ computably enumerates this sequence then $h$ defines a reduction $\text{Id} \leq_c R$. Therefore, $\text{Id} \equiv_c R$ by minimality of $R$.

Suppose now that $R \equiv_c \text{Id}$ or, for every c.e. set $W$, if $W$ hits infinitely many $R$-classes then it hits each of them. If $R \equiv_c \text{Id}$ then we have nothing to prove, so we suppose we are in the latter case. If a ceer $S$ has infinitely many classes and $S \leq_c R$ via some computable function $f$ then the range $W$ of $f$ hits infinitely many $R$-classes, and, therefore $\text{im}(f)$ hits each $R$-class, i.e. $S \leq_c R$ is an onto-reduction. Hence, $R \leq_c S$.

**Theorem 6.3.** There is an infinite $\leq_c$ anti-chain of weakly precomplete minimal ceers.

**Proof.** Andrews and Sorbi \cite{AndrewsSorbi} Theorem 3.3] showed that there are infinitely many incomparable minimal dark ceers. They proceed to build these ceers $E_i$ for $i \in \omega$ via a finite injury argument where each requirement may cause some collapses (respecting the restraint placed by higher priority requirements) and may place a finite restraint, i.e., there are finitely many triples $(a, b, j)$ which represent that $a$ and $b$ are restrained from becoming $E_j$-collapsed by lower priority requirements.

We need only note that we can add a requirement of type $\text{WP}_i^j$ to ensure that $E_i$ has a $\varphi_j$-fixed point within this framework. These requirements need to place no restraint, and they obey such restraints as long as the witness $x$ chosen is distinct from any of the restrained classes.

\[ \Box \]

**References**


