THE PUNCTUAL DEGREE STRUCTURE OF EQUIVALENCE RELATIONS

HEER TERN KOH AND KENG MENG NG

Abstract. We provide a full characterisation of which equivalence relations have a dense punctual degree structure.

1. Introduction

Computable structure theory is an area of research that studies the algorithmic content of mathematical structures. The main objects of interest are *computable structures*, which are defined as structures with domain $\mathbb{N}$ and having uniformly computable operations and relations [15, 12]. A large body of work in computable structure theory is dedicated to studying isomorphisms between computable structures. If there exists a computable isomorphism between any two isomorphic computable copies of a structure A, then we say that the structure A is *computably categorical*. A well-known example of such a structure is the countable dense linear order with no endpoints. In some sense, being computably categorical reflects the algorithmic nature of the classical proof of its categoricity. It should not be too surprising that not all structures are computably categorical. For such structures, there have been some interest in studying Δ^0_n*-categoricity*, which informally speaking, measures how much non-computable information is encoded by the isomorphisms of such structures.

Rather than measuring the amount of non-computable information required to compute isomorphisms of certain structures, a recent research program seeks instead to measure the amount of unbounded search required [8]. More broadly, the research program seeks to investigate the question, "What happens if we forbid unbounded search in computable structure theory?" To investigate this question, it was proposed that one should study *punctual structures* defined as follows.

Definition 1.1 ([8])**.** *A countable structure is punctual if its domain is* $\mathbb{N}$ *and its operations and relations are uniformly primitive recursive.*

Under this framework, it follows intuitively that the isomorphisms should also be similarly restricted to those computable without delay. However, as the inverse of a primitive recursive function is not necessarily primitive recursive, it is not immediately obvious what the correct notion of isomorphisms between punctual structures should be. Within the literature, there have been some work on studying punctual structures up to different notions of isomorphisms like punctual isomorphisms and honest isomorphisms [9, 14, 4]. Another approach is to think of the relation "being primitive recursively isomorphic" as a reduction rather than a symmetric relation [9]:

Date: June 4, 2024.

K.M. Ng was supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (MOE-T2EP20222-0018).

Definition 1.2 ([8]). *Given two isomorphic punctual structures A and B we define $A \leq_{pr} B$ if there exists a primitive recursive isomorphism $f : A \to B$. This is a pre-ordering of the set of all punctual copies of a fixed punctual structure $\mathcal{A}$. The equivalence relation induced by $\leq_{pr}$ is known as the punctual degrees of $\mathcal{A}$.*

The basic idea of considering the partial ordering of the punctual degrees of $\mathcal{A}$ is that it allows formal insight into the punctual content of the structure $\mathcal{A}$. If $A \leq_{pr} B$ then it intuitively means that B is revealed at least as fast as A (up to a punctual delay). By considering how all punctual copies of $\mathcal{A}$ relate to each other under $\leq_{pr}$, it allows a different way of understanding how easily the algebraic aspects of $\mathcal{A}$ can be concealed. For instance, having a greatest punctual degree means that the structure $\mathcal{A}$ has a "fastest" copy in which all important information of $\mathcal{A}$ must be revealed quickly. Having a least punctual degree means that the structure $\mathcal{A}$ has a "slowest" copy which produces no more than the absolutely necessary information, for instance finitely generated free groups.

The study of the induced degree structure has collected a number of interesting results. For example, the punctual degrees of the countable dense linear ordering with no endpoints, the random graph and the universal countable abelian p-group have been shown to be pairwise non-isomorphic [13], suggesting that the algorithmic content encoded by the respective isomorphisms differ on some sub-recursive level. This contrasts sharply with the fact that these structures are largely indistinguishable when viewed through the coarser lense of computable isomorphisms.

When faced with a degree structure (as a partial ordering), one of the immediate questions is whether the structure is bounded (from above or below), and whether or not the degree structure is dense. Indeed, some of the early work in classical degree theory focussed on the density of various degree structures. Some well-known results that come to mind include the results of Spector (global Turing degrees), Sacks (c.e. Turing degrees), Cooper (local enumeration degrees) and Gutteridge (global enumeration degrees). It is therefore absolutely natural to classify the punctual structures which have dense punctual degree structures.

Apart from the traditional interest in density, this question is also important in that it allows us to understand how finely stratified the different relative speed of enumerations of $\mathcal{A}$ are. For instance, it has been shown that algebraically simple classes like finitely generated structures, or (computably almost) rigid structures have dense punctual degrees [1, 3]. This is due to the fact that in these classes we can vary the speed of enumerating the structure using the relatively simple descriptions. In fact, Kalimullin, Melnikov and Zubkov [10] showed that every countable distributive lattice can be embedded in the punctual degrees of a rigid finitely generated structure. In contrast, Melnikov and Ng [14] produced a structure of punctual dimension 2, where the structure has exactly two different speeds of enumeration.

Proving that a structure *does not* have a dense degree structure turned out to be a much more difficult problem. In fact no such degree structure was known until relatively recently. The first example produced in [6] was that of a graph. However, one could argue that this example was a pathological one specifically constructed to have such a property. Remarkably, it was later shown that the punctual degrees of the countable dense linear ordering without endpoints is not dense [11]. This suggests that the punctual degrees of even relatively simple relational structures can potentially possess non-trivial properties. The aforementioned result is also surprising in that the given structure is a very natural one which is homogeneous, and therefore we cannot encode

information in the usual way, nor can we identify "gadgets" or "components" that a computability-theoretic proof might do.

In this paper, we continue with the systematic study of the punctual degrees of various natural classes of structures. In particular, we study the density of the punctual degrees of equivalence structures. We fully classify the equivalence relations that generate dense punctual degree structures, using a relatively simple condition:

Theorem 2.5. *Let E be a punctual equivalence relation. The punctual degrees of E is not dense iff there exists nice (to be defined later) functions G_1, G_2 such that for all $y \in \omega$, either $G_1(y) < G_2(y)$ or $G_2(y) = \infty$.*

As shown in [2, 7, 5], isomorphisms between equivalence structures can be algorithmically complicated; isomorphisms between certain computable equivalence structures can only be found using a Δ^0_3 oracle (folklore). Much of this complexity comes from computing the sizes of each class. In some ways, the result above reflects this intuition. The existence of the nice functions provides an algorithmically complex set of sizes within the equivalence structures, resulting in pathological properties in the respective punctual degrees. However, even when the set of sizes is seemingly simple, we may still obtain non-density. As shall be shown, having infinitely many infinite classes is a sufficient condition for non-density of the punctual degrees of equivalence structures (recall that the equivalence structure with only infinite classes is computably categorical).

The techniques used in this paper is a culmination of various techniques discovered thus far. In particular, the proof strategy for non-density combines techniques from the first example of non-density and the aforementioned result about the countable dense linear ordering with no endpoints. We believe that the proof presented here is a nice middle ground between the two; we do not have the *pressing* advantage as the isomorphism type is fixed beforehand, but also manage to avoid the high combinatorial complexity in the case for the dense linear order. In this way, we hope that this paper provides a clearer picture of the essence of non-density of the punctual degrees. Furthermore, we hope the characterisation presented in this paper is a step towards meta-theorems relating algebraic properties of structures with degree-theoretic properties of its punctual degrees.

Nevertheless the proof of the main result of this paper is still rather delicate. This is particularly true in the case where we analyse the equivalence relations that do not have a dense punctual degree structure. As mentioned above, the main difficulty is that even in these cases, the structure is a natural one in which the usual encoding methods do not work. This forces the complexity of the proof to shift towards the combinatorial aspects and analysis of the strategies.

2. Definitions and the main idea

Definition 2.1. *Let $E, F \subseteq \omega^2$ be equivalence relations. We say that $E \leq_{pr} F$ iff there is some bijective primitive recursive function $h : \omega \to \omega$ such that for any $m, n \in \omega$,*

$$(m, n) \in E \iff (h(m), h(n)) \in F.$$

If we also have that $F \not\leq_{pr} E$, then we write $E <_{pr} F$.

We think of E, F as primitive recursive structures (ω, E) and (ω, F). Consequently, each primitive recursive function h can be interpreted as a map between structures, written as $h : E \to F$. Similarly, we will also write $m \in E$ to mean that m is an element of the domain of (ω, E).

Notation 2.2. *Let $E \subseteq \omega^2$ be a punctual equivalence relation, and let $m, n \in \omega$ be given.*

- *We write $m =_E n$ iff $(m, n) \in E$.*
- *We use $[m]_E$ to denote the equivalence class of E containing m.*
- *We use $\#[m]_E^s$ to denote the size at stage s of $[m]_E$. Similarly, $\#[m]_E$ denotes the size of $[m]_E$ in the limit.*

Definition 2.3. *Let $G : \omega^2 \to \omega$ be a total computable function. For a natural number N we say that G is N-nice for E if G has the following properties.*

- *For all $y, s \in \omega$, $G(y, s) \leq G(y, s + 1)$.*
- *For all $y \in \omega$, $\sup_s G(y, s) > N$.*
- *For all $y \in \omega$, there are at least y classes of size exactly $\sup_s G(y, s)$ in E. This condition must still hold even when $\sup_s G(y, s) = \infty$.*

To avoid making the notation heavy, we simply write $G(y)$ to mean $\sup_s G(y, s) \in \omega \cup \{\infty\}$. Furthermore, although the definition of 'nice-ness' depends on E, since we fix E before a construction, we simply call a function N-nice. If in addition $N = 0$, we then say that G is nice.

Lemma 2.4. *There exists an N-nice G iff there exists an N-nice G that is non-decreasing[1] in y; for all $y \in \omega$, $\sup_s G(y, s) \leq \sup_s G(y + 1, s)$.*

Proof. The converse is trivial. Suppose that there is some N-nice G. We split the proof non-uniformly into two cases.

If there exists some n for which there are infinitely many y such that $G(y) = n$, then define $F(y) = n$ for all y. F is clearly a non-decreasing N-nice function.

If for each n, there are only finitely many y such that $G(y) = n$, then define F recursively as follows. $F(2)$ to be the first place where G is finite, and $F(y + 1) = G(x)$ where $x \in \omega$ is the least such that $x > y$ and $G(x) \geq F(y)$. Such an x must exist as there can only be finitely many x' for which $G(x') < F(y)$. It is also evident that F as defined is a non-decreasing N-nice function. □

We present a classification for density of the punctual degrees of E as follows.

Theorem 2.5. *Let E be a punctual equivalence relation. The punctual degrees of E is not dense iff there exists nice non-decreasing (in y) functions G_1, G_2 such that for all $y \in \omega$, either $G_1(y) < G_2(y)$ or $G_2(y) = \infty$.*

The proof of the theorem will be split into the following lemmas.

Lemma 2.6. *Let E be a punctual equivalence relation with only finitely many infinite classes and where there is at most one size which is repeated infinitely often.*

- *If there are no sizes repeated infinitely often and there does not exist a nice G, or*

[1] We adopt the usual convention that for any $n \in \omega$, $n < \infty$.

- *there is exactly one size N repeated infinitely often and there does not exist a N-nice G,*

then the punctual degrees of E is dense.

Lemma 2.7. *Let E be a punctual equivalence relation. If there exists nice non-decreasing G_1, G_2 such that for all y, either $G_1(y) < G_2(y)$ or $G_2(y) = \infty$, then the punctual degrees of E is not dense.*

Proof of Theorem 2.5. The converse can be obtained directly from Lemma 2.7. It remains to argue that the forward direction follows from Lemma 2.6.

Fix some punctual equivalence relation E. First suppose that there are no nice non-decreasing functions G_1, G_2 such that for all y, either $G_1(y) < G_2(y)$ or $G_2(y) = \infty$. Then we wish to apply Lemma 2.6 to conclude that E is dense. Observe that E can only have finitely many infinite classes, otherwise the functions $G_1(y) = G_2(y) = \infty$ for all y, would be nice functions contradicting the assumption. Furthermore, if E has at least two different sizes repeated infinitely often, say $M_1 < M_2 < \infty$, then $G_1(y) = M_1$ and $G_2(y) = M_2$ for all $y \in \omega$ would also serve as nice functions contradicting the assumption. Thus E must have only finitely many infinite classes and at most one size repeated infinitely often. Now consider the following two cases; E has one size N repeated infinitely often or E has no sizes repeated infinitely often.

If E has exactly one size N repeated infinitely often, by Lemma 2.6, we would be done if we can show that there does not exist an N-nice G. Suppose for a contradiction that there exists a N-nice G. Applying Lemma 2.4, we may assume that G is a non-decreasing N-nice function. Then $G_1(y) = N$ and $G_2(y) = G(y)$ gives two nice non-decreasing G_1, G_2 for which $G_1(y) = N < G_2(y)$ for all y, a contradiction. If on the other hand, E has no sizes repeated infinitely often, and there exists a nice G, then it can easily be made strictly increasing by following the proof of Lemma 2.4. Now we can define non-decreasing nice G_1, G_2 by letting $G_1(y) = G(2y)$ and $G_2(y) = G(2y+1)$, a contradiction. □

The rest of this paper will be dedicated to proving Lemma 2.6 in Section 3 and Lemma 2.7 in Section 4.

3. Proof of Lemma 2.6

Lemma 2.6. *Let E be a punctual equivalence relation with only finitely many infinite classes and where there is at most one size which is repeated infinitely often.*

- *If there are no sizes repeated infinitely often and there does not exist a nice G, or*
- *there is exactly one size N repeated infinitely often and there does not exist a N-nice G,*

then the punctual degrees of E is dense.

Let $B \cong T$ punctual equivalence relations be given such that $B <_{pr} T$. We also assume that B, T satisfies the premise of Lemma 2.6; B, T have only finitely many classes of infinite size and at most one size which is repeated infinitely often. Let $h : B \to_{onto} T$ be the primitive recursive isomorphism which witnesses $B \leq_{pr} T$ and fix some listings $\{\beta_e\}_{e\in\omega}$ and $\{\alpha_e\}_{e\in\omega}$ of the primitive

recursive functions. We construct A, an equivalence relation, with primitive recursive isomorphisms $p : B \to_{onto} A$ and $q : A \to_{onto} T$ such that $qp = h$, while satisfying the following requirements.

$$\begin{aligned} P_e : &\quad \beta_e : A \to B \text{ is not a surjective isomorphism.} \\ Q_e : &\quad \alpha_e : T \to A \text{ is not a surjective isomorphism.} \end{aligned}$$

Let $N < \infty$ be the size which is repeated infinitely often by B, T. To make the description uniform, if there are no sizes which are repeated infinitely often, then we take $N = 0$. In addition, we may also assume that N is the least size repeated infinitely often in B, T and that there are no classes of smaller size in B, T. This is because there can only be finitely many classes of smaller sizes, on which we can non-uniformly fix our definitions of p, q respectively.

3.1. **Informal description.** During the construction, given $\beta_e : A \to B$, we will monitor if for any $a, a' \in A$, $\beta_e(a) =_B \beta_e(a')$ iff $a =_A a'$. If ever we find some a, a' such that this does not hold, then β_e cannot possibly be an isomorphism and we need not act anymore for this requirement. We also monitor a similar property for $\alpha_e : T \to A$; for any $t, t' \in T$, $\alpha_e(t) =_A \alpha_e(t')$ iff $t =_T t'$. These assumptions will be implicit throughout the rest of Section 3.

3.1.1. *Main idea.* Each requirement will attempt to define some N-nice G. In doing so, each requirement can only be attended to for a finite amount of time, otherwise we would succeed in defining the N-nice function, contradicting the assumption of Lemma 2.6.

Since each requirement will be attended to for only a finite amount of time and essentially do not interact, we suppress the indices until the formal construction in Section 3.2. While attempting to satisfy some P-requirement, we will have A 'copy' the structure T by making q^{-1} look temporarily primitive recursive. Then we can attempt to diagonalise against $\beta : A \to_{onto} B$ by defining some $\varphi : T \to_{onto} B$ based on β. Evidently, such a φ cannot be successfully defined or else $B \equiv_{pr} T$. Thus, either β is observed to fail in finite time or it has to be temporarily non-surjective. Whenever we find some potential witness for the non-surjectivity of β, we will abandon the current definition of φ, and instead extend the definition of the N-nice function. While waiting for β to 'recover' by providing a preimage to the potential witness, we attend to other requirements in the meantime. The idea is that if β attempts to 'recover' from non-surjectivity infinitely often, then we will succeed in defining the N-nice function on all inputs $y \in \omega$. The Q requirements will be satisfied in a similar way; instead of making q^{-1} look primitive recursive, we make p^{-1} look primitive recursive and attempt to define $\varphi : T \to_{onto} B$ based on $\alpha : T \to_{onto} A$.

3.1.2. *Strategy for P.* In order to prepare for the strategy for P, we start by letting A 'copy' T; ensure that $q^{-1}(t)\downarrow$ for each t currently in T. Once we have that $A_s \cong T_s$, we may begin the strategy for P. While attending to P, we continue ensuring that $q^{-1}(t)\downarrow$ for each newly enumerated t as follows.

- If there exists some $t' \in T_s$ such that $t' =_T t$ and $q^{-1}(t')\downarrow = a'$, then enumerate an element a into A_s, and define $a =_A a'$, $q(a) = t$.
- If there is no such $t' \in T_s$ where $t' =_T t$ and $q^{-1}(t')\downarrow$, then enumerate an element a into A_s where $a \neq_A x$ for any $x \in A_s$ and let $q(a) = t$.

In addition, we may then define $p(b) = q^{-1}h(b)$, ensuring that $qp = h$ and that p is kept primitive recursive.

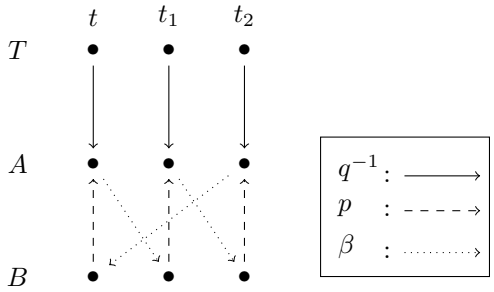


FIGURE 1. β-locus

In order to satisfy P, we attempt to define $\varphi : T \to_{onto} B$ based on β. More specifically, for each $t \in T$, $\varphi(t) =_B \beta(p\beta)^k q^{-1}(t)$ for some $k \in \mathbb{Z}$. Obviously we cannot possibly succeed in doing so as $B <_{pr} T$. Our attempt at defining φ fails when we find some $s \in \omega$ and $t \in T$ such that $\#[t]_T$ stays strictly larger than $\#[\varphi(t)]^s_B$ for longer than φ is allowed to wait. When this is discovered, we will abandon this definition of φ and instead extend the definition of G. For the rest of this discussion, we fix $y \geq 2$ such that $G(x)$ has been defined for each $x < y$.

We first address how we will keep φ primitive recursive and injective. Given some $t \in T_s$ which is not yet in the domain of φ, if there is some $t' \in T_s$ such that $\varphi(t')\downarrow$ and $t =_T t'$, then map $\varphi(t)$ to some element in $[\varphi(t')]_B$ not yet in $\mathrm{rng}(\varphi)$. If such an element does not exist, that is, it currently looks like $\#[t']^s_T > \#[\varphi(t')]^s_B$, then we abandon this attempt at defining φ. If there is instead no $t' \in T_s$ for which $t' =_T t$ and $\varphi(t')\downarrow$, then search for the least $k \geq 0$ for which $b \coloneqq \beta(p\beta)^k q^{-1}(t)$ is such that $b \neq_B b'$ for any b' currently in the range of φ. Once such a b is found, then let $\varphi(t) = b$. This ensures that φ does not map two distinct equivalence classes in T to the same one in B. We now provide a brief explanation as to why such a k can always be found quickly. (For the formal proof of this, we refer the reader to Lemma 3.6.)

Definition 3.1. *For $t, t' \in T$, let $t \sim t'$ if there exists some $k \in \mathbb{Z}$ such that $q^{-1}(t') =_A (p\beta)^k q^{-1}(t)$.*

The relation $\sim$ as defined above is clearly an equivalence relation on the elements in $\mathrm{rng}(q)$. We shall refer to the equivalence class $[t]_\sim$ as the β-locus of t. In addition, if there is some $k > 0$ such that $q^{-1}(t) =_A (p\beta)^k q^{-1}(t)$, then we say that the β-locus of t *closes*.

Given some $t \notin \mathrm{dom}(\varphi)$, and $t \neq_T t'$ for any t' currently in $\mathrm{dom}(\varphi)$, for each $0 \leq i \leq |B_s| + 1$, check if $\beta(p\beta)^i q^{-1}(t) =_B b$ for some b currently in the range of φ. If the β-locus of t has yet to close, then there must be an image available for $\varphi(t)$ as $\beta(p\beta)^{|B_s|+1} q^{-1}(t) \notin B_s$ and thus not in the range of φ. On the other hand, if the β-locus has already closed, then the number of T-classes in $[t]_\sim$ must be the same as the number of B-classes in $\{\beta q^{-1}(t') \mid t' \in [t]_\sim\}$ as q^{-1} and β must currently be injective on classes (see Fig. 1 for an illustration). Provided that φ is currently well-defined on classes, it follows that there is at least one B-class in $\{\beta q^{-1}(t') \mid t' \in [t]_\sim\}$ that is not yet in $\mathrm{rng}(\varphi)$ (see Lemma 3.6 for the details).

The procedure described above guarantees that φ is primitive recursive and injective on both elements and classes provided that we never abandon its definition (recall that this means that for all s and for all $t \in T_s$, $\#[t]^s_T \leq \#[\varphi(t)]^s_T$). If β is surjective, note that φ as defined above will also

be surjective. However, since surjectivity is a Π^0_2 property, we will only discover whether or not β is surjective in the limit. Thus, we need to actively attempt to make φ surjective regardless of the current behaviour of β. Recall that the plan is to successfully define φ, or $G(y)$, or to discover that β is not an isomorphism. More specifically, in the event that we are unable to keep φ primitive recursive and injective or we find that φ cannot currently be made surjective, we need to progress the strategy for P by finding at least y many T-classes of sizes $> N$.

Let $b \in B$ be an element that is currently not in the range of φ. If the β-locus of $h(b)$ closes, then b must enter $\mathrm{rng}(\varphi)$ shortly after. Although we do not require b to enter the range of φ quickly to satisfy surjectivity requirements for φ, it is important that it does so here. This is because we have not made any progress in defining $G(y)$ and cannot simply wait without any guarantee that β will be surjective. We may thus assume that the β-locus of $h(b)$ does not close. Further wait for some stage s such that the following holds.

- There are at least y many T-classes in $[h(b)]_\sim$.
- $\#[b]^s_B \geq N$.

Since the β-locus is assumed not to close and N is assumed to be the smallest class size, such a stage s is guaranteed to exist. Once such a stage is reached, we may then attempt to place b in $\mathrm{rng}(\varphi)$ as follows.

Case 1: If there does not currently exist any t' such that $\varphi(t') =_B b$, then pick some $t \sim h(b)$ such that $t \neq_T t'$ for any t' currently in $\mathrm{dom}(\varphi)$ and define $\varphi(t) = b$.

Case 2: If there currently exists some t' such that $\varphi(t') =_B b$, then wait for a stage s' such that $\#[t']^{s'}_T \geq N$. Once again, since N is assumed to be the smallest class size in T, such a stage is guaranteed to exist. At such a stage, b must have entered $\mathrm{rng}(\varphi)$ unless $\#[b]^{s'}_B > \#[t']^{s'}_T \geq N$. If this happens, we cannot wait for some later stage s'' where $\#[t']^{s''}_T \geq \#[b]^{s'}_B$ as there is no guarantee that such a stage will exist. Instead, we can now define $G(y) = \#[h(b)]_T > N$ and abandon the definition of φ.

Observe that in both cases, when we abandon the definition of φ, we find some t where there are at least y many T-classes in $[t]_\sim$ and $\#[t]_T > N$. Furthermore, if we ever find some T-class such that $\#[t]^s_T > \#[\varphi(t)]^s_B$, it must be that $\varphi(t) =_B \beta(p\beta)^k q^{-1}(t)$ for some $k < 0$ and thus must have been defined via Case 1 above. Recall that when defining φ for injectivity requirements, we always have that $k \geq 0$. That is to say, if we ever have to abandon the definition of φ, we are able to find some β-locus which contains at least y many T-classes with at least one of them, say t, having size $> N$ and can thus define $G(y) = \#[t]_T$. Once we have defined $G(y) = \#[t]_T$, wait for the β-locus of t to close. Such a wait may never terminate, but this implies that β cannot be surjective, otherwise there will be infinitely many T-classes in $[t]_\sim$ all of the same size $> N$, which is impossible. Thus, we either wait forever and satisfy P, or there is some finite stage at which the β-locus closes. While waiting, we will attend to other requirements, and only return to this requirement when the β-locus of t closes.

Once the β-locus of t closes, provided that β is an isomorphism, all T-classes within $[t]_\sim$ will eventually have the same size. That is, either $G(y)$ will be correct or β fails to be an isomorphism. We then begin a new attempt at defining φ as described thus far. In summary, under the assumption that β is a surjective isomorphism, the strategy for the P-requirement has the following outcomes.

- Every version of φ that we begin must be abandoned, otherwise $\varphi : T \to_{onto} B$ is a primitive recursive isomorphism (see Lemma 3.6).
- We can only finitely often abandon and begin defining some version of φ, otherwise $G(x)$ is successfully defined (see Lemma 3.8).
- If we wait forever for some β-locus to close, then β cannot be surjective, otherwise there are infinitely many T-classes of the same size $> N$ (see Lemma 3.9). Thus, P must be satisfied without being returned to after some finite stage.

3.1.3. *Strategy for Q.* The strategy to satisfy Q will be rather similar to the strategy for P. We summarise the ideas needed for the strategy to work.

- A 'fast' map from T to A, provided previously via q^{-1}.
- A 'fast' map from A to A, provided previously via $p\beta$.
- A 'fast' map from A to B, provided previously via β.

By taking the compositions of the maps above, we are able to define $\varphi : T \to B$. When attempting to satisfy Q, observe that we already have 'fast' maps $\alpha : T \to A$ and $\alpha q : A \to A$. The final ingredient needed is to ensure that $p^{-1} : A \to B$ is 'fast', at least for the stages during which we are attending to Q. This matches the intuitive idea that in making A resemble B, we can use the assumption that $B <_{pr} T$ to obtain a diagonalisation against $\alpha : T \to A$.

To make A 'copy' B, we need to slow down the enumeration of A while keeping it primitive recursive. Stop enumerating any new elements into A unless some element b is enumerated into B satisfying one of the following.

- If there does not exist a' currently in A such that $q(a') =_T h(b)$, then enumerate a new element a into A and let $p(b) = a$ and $q(a) = h(b)$.
- If there is some a' currently in A such that $q(a') =_T h(b)$ but $\#[a']_A < \#[b]_B$, then let $\#[a']^s_A = \#[b]^s_B$ for stages s where Q is being attended to.

In other words, we enumerate new elements into A only for the sake of maintaining $p : B \to A$. As long as the definition of p is not in danger of failing, we never enumerate elements into A. To see that A remains primitive recursive, observe that at each stage, there can only be primitive recursively many elements in T but not in B. Therefore, after some primitive recursively bounded number of enumerations of elements into B, we must also enumerate some new element into A.

Once $p^{-1}(a)\downarrow$ for every a currently in A, then we may begin our attempt to define $\varphi : T \to B$ in a similar way as before; for each $t \in T$, $\varphi(t) =_B p^{-1}(\alpha q)^k \alpha(t)$ for some $k \in \mathbb{Z}$. Just as described previously, as long as α is primitive recursive and injective, we will be able to maintain φ to also be primitive recursive and injective. Furthermore, each potential witness of non-surjectivity of φ (and hence α), will be paired with an extension of the definition of G. Either some true witness is found, in which case α cannot be surjective, or we succeed in defining G if Q is attended to infinitely often.

3.2. **Formal construction.** As discussed in Sections 3.1.2, 3.1.3, the strategies for P and Q are essentially the same. To simplify the construction, we relabel the requirements as R_e, where $R_{2e} = P_e$ and $R_{2e+1} = Q_e$. The requirements are then arranged in order of priority as $R_0, R_1, R_2, \ldots$. For the sake of each requirement R_e, we also make the following definitions.

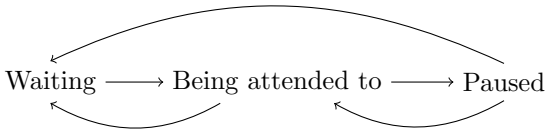


FIGURE 2. Transitions between states

Definition 3.2. *For each $e \in \omega$, define the following.*

- $f_e = \begin{cases} p\beta_n & \text{if } e = 2n \\ \alpha_n q & \text{if } e = 2n+1 \end{cases}$
- $r_e = \begin{cases} q^{-1} & \text{if } e = 2n \\ \alpha_n & \text{if } e = 2n+1 \end{cases}$
- $d_e = \begin{cases} \beta_n & \text{if } e = 2n \\ p^{-1} & \text{if } e = 2n+1 \end{cases}$

The idea here is that while we are attending to R_e, the maps with index e will behave primitive recursively.

For the sake of each R_e, we will attempt to define either $\varphi_e : T \to B$ or $G_e \coloneqq \lim_s G_{e,s}$ a N-nice function. In the informal description, when attempting to define $\varphi_e : T \to B$, we utilise an equivalence relation $\sim$. Here we provide a more general definition to be used later in the formal construction.

Definition 3.3. *Let $f_e : A \to A$ and $r_e : T \to A$ be isomorphisms. Then define a relation $\sim_e$ on T such that $t \sim_e t'$ if there exists some $k \in \mathbb{Z}$ such that $f_e^k r_e(t) =_A r_e(t')$.*

It is clear that the relation is an equivalence relation and we call each equivalence class $[t]_{\sim_e}$ the f_e-locus of t. In addition, if there is a positive k such that $f_e^k r_e(t) =_A r_e(t)$, then we say the f_e-locus of t has closed.

Throughout the construction, each R_e will always be in one of the following states.

(1) Waiting.

(2) Being attended to.

(3) Paused.

(4) Satisfied.

At any given stage, only one requirement will be in state being attended to. Whenever some requirement enters the state being attended to, all lower priority requirements not yet in the satisfied state goes back to the waiting state. The state of a requirement (which has yet to enter the satisfied state) possibly goes through the changes as illustrated in Fig. 2. For each requirement R_e, we maintain a marker $y_{e,s}$ defined stage-wise in order to track the inputs on which $G_{e,s}$ is already defined. While a requirement R_e is being attended to, we attempt to define $\varphi_e : T \to B$. A

requirement that is being attended to becomes paused at some stage s if there exists some $b \in B_s$ such that all of the following holds. (We shall refer to the conjunction of all of the following properties as $(**)$.)

- $b \notin \mathrm{rng}(\varphi_e)$.
- There is some $t \in T_s$ such that $t \sim_e h(b)$ and $\#[t]^s_T > N$.
- There are at least $y_{e,s}$ distinct T-classes in $[h(b)]_{\sim_e}$.

If such an element exists, we denote it as b^e. For a paused requirement, we say that it is *ready* at stage s if the f_e-locus of $h(b^e)$ has closed.

Throughout the construction, we also assume that the following actions automatically take place.

A1. If we find $a, a' \in A_s$ such that $a \neq_A a'$ but $\beta_e(a) =_B \beta_e(a')$, or $a =_A a'$ but $\beta_e(a) \neq_B \beta_e(a')$, then we change the state of R_{2e} to satisfied.

A2. If we find $t, t' \in T_s$ such that $t \neq_T t'$ but $\alpha_e(t) =_A \alpha_e(t')$, or $t =_T t'$ but $\alpha_e(t) \neq_A \alpha_e(t')$, then we change the state of R_{2e+1} to satisfied.

A3. Once we have defined $\varphi_e(t)$ for some $t \in T_s$, for any other t' entering T in subsequent stages such that $t' =_T t$, map $\varphi_e(t')$ to some element in $[\varphi_e(t)]_B$ not yet in the range of φ_e. If we find that no such suitable element in $[\varphi_e(t)]_B$ exists, then abandon this definition of φ_e and change the state of R_e to paused. Furthermore, let $b^e = \varphi_e(t)$, and resume the requirement when the f_e-locus of $h(b^e)$ has closed.

A4. Whenever some element b enters B_s, define $p(b) = q^{-1}h(b)$ if it currently exists in A. Otherwise, enumerate a new element a into A_s and define $p(b) = a$ and $q(a) = h(b)$.

We are now ready to present the formal construction.

Stage 0**:** For each $e \in \omega$ define $y_{e,0} = 2$ and let $G_{e,0}(n) = 0$ for all $n \in \omega$.

Stage $s > 0$**:** Search for the least e such that one of the following holds.

(1) R_e is currently paused but ready to be resumed.

(2) R_e is waiting.

(3) R_e is being attended to.

In each of the above cases, do the following.

Case 1: Change all lower priority requirements not in the satisfied state to the waiting state. For each $e' > e$, let $G_{e',s}(n) = 0$ for all $n \in \omega$ and also let $y_{e',s} = 2$. Recall that if a requirement was paused and becomes ready to resume, then the f_e-locus of $h(b^e)$ has closed. Restart our attempt at defining φ_e by letting $\varphi_e(t) = d_e r_e(t)$ for each t such that the f_e-locus of t is closed. In particular, we now have that $b^e \in \mathrm{rng}(\varphi_e)$. For the remaining $t \in T_s$, we refer to Case 3.

Case 2: Change the state of R_e to being attended to and all lower priority requirements not in the satisfied state to the waiting state. Now refer to Case 3 for the actions required when R_e is being attended to.

Case 3: If R_e is a P-requirement ($e = 2n$ for some $n \in \omega$), then let A 'copy' T. That is, for each $t \in T_s$, if $q^{-1}(t)\uparrow$, enumerate an element a into A_s and let $q(a) = t$. If R_e is a Q requirement ($e = 2n + 1$ for some $n \in \omega$), then wait for B to 'catch up' with A. As long as there exists some $a \in A_s$ such that $p^{-1}(a)\uparrow$, we do nothing (except the automatic actions A1-A4) and proceed to the next stage.

We may now assume that the functions $r_e : T \to A$, $f_e : A \to A$ and $d_e : A \to B$ are all temporarily primitive recursive. While R_e is being attended to, we keep φ_e primitive recursive and attempt to make it surjective. If the current stage is even, proceed to Case 3i, and proceed to Case 3ii otherwise.

Case 3i: During even stages s, let $t \in T_s$ be the lowest indexed element not currently in the domain of φ_e. We may assume that there are currently no other $t' \in T_s$ such that $\varphi_e(t')\downarrow$ and $t' =_T t$, otherwise Action A3 would have defined $\varphi_e(t)$ or paused R_e. Search for the first $k \geq 0$ such that $d_e f_e^k r_e(t) \neq_B b$ for any b currently in $\mathrm{rng}(\varphi_e)$. Once found, define $\varphi_e(t)$ to be the smallest indexed element in $[d_e f_e^k r_e(t)]_B$. We check in Lemma 3.6 that such a class can always be found in primitive recursive time.

Case 3ii: During odd stages s, we attempt to make φ_e surjective. Let $b \in B_s$ be the lowest indexed element satisfying all of the following. (We shall refer to the conjunction of all of the following properties as $(*)$.)

- b is not currently in the range of φ_e.
- $\#[b]_B^s \geq N$.
- There are at least $y_{e,s}$ many distinct T-classes in the f_e-locus of $h(b)$.

If there is also some $t \in \mathrm{dom}(\varphi_e)$ such that $\varphi_e(t) =_B b$, then we wait for some stage $s' \geq s$ where either $\#[b]_B^{s'} > N$ or $\#[t]_T^{s'} \geq N$. At such a stage, if the former holds, then observe that b satisfies the conditions $(**)$, and can thus pause the requirement, let $b^e = b$ and proceed to attend to other requirements. If on the other hand $\#[b]_B^{s'} = N$, and $\#[t]_T^{s'} \geq N$, then b must have been placed in $\mathrm{rng}(\varphi_e)$ via Action A3.

At any stage s, if R_e is to be paused, then abandon the current definition of φ_e and define the following. $y_{e,s+1} = y_{e,s} + 1$ and $G_{e,s'}(y_{e,s})$ for each $s' > s$ to be $\#[h(b^e)]_T^{s'}$. (Recall that b^e is defined whenever R_e becomes paused.) When proceeding to the next stage, let $G_{e',s+1} = G_{e',s}$ and $y_{e',s+1} = y_{e',s}$ for any e' unless otherwise stated in one of the above actions.

3.3. **Verification.**

Lemma 3.4. *$q : A \to T$ is a primitive recursive isomorphism and $qp = h$.*

Proof. We proceed by induction on the elements in their order of enumeration into A. The base case is trivial. Suppose recursively that q is a primitive recursive isomorphism on the first n many elements in A. When the $n+1$-th element, a, is enumerated, we consider the following possibilities.

- A is currently 'copying' T. Recall from Section 3.1.2 that when this is happening, for each t currently in T, if $q^{-1}(t)$ is undefined, we enumerate some element into A to serve as the q

preimage for t. That is, when a is enumerated, there must already exist some q image for a. Furthermore, such an image is chosen carefully to ensure that q is an isomorphism.

- A is currently 'copying' B. We refer the reader to Section 3.1.3. During such stages, elements will only be enumerated into A if some element b is enumerated into B and $q^{-1}h(b)$ is currently undefined. Then depending on whether there are currently a' such that $q(a') =_T h(b)$, the new element a will be enumerated into either an existing A-class or a new A-class to ensure that q is a primitive recursive isomorphism.

Observe that the above actions also enusre that $qp = h$. □

A similar analysis will allow us to obtain that p is also primitive recursive. Applying this fact with the lemma above gives us the following.

Corollary 3.5. *$p : B \to_{onto} A$ and $q : A \to_{onto} T$ are primitive recursive surjective isomorphisms.*

To prove that each R_e is satisfied, we follow a standard priority argument. The idea is as follows. First, we prove that each R_e can only be attended to for finitely many stages (see Lemmas 3.6 and 3.8). Next, we prove that if we never again attend to R_e, then it must be satisfied (Lemma 3.9).

Lemma 3.6. *If R_e is attended to for cofinitely many stages, then $\varphi_e : T \to B$ is a surjective primitive recursive isomorphism.*

Proof. By assumption that R_e is attended to for cofinitely many stages, there is some version of φ_e that is never abandoned. We prove that this attempt at defining φ_e produces a primitive recursive surjective isomorphism. Given some element $t \in T$, $\varphi(t)$ could have been defined in one of the following ways. Suppose inductively that φ_e currently satisfies the property that $\varphi_e(x) =_B \varphi_e(x')$ iff $x =_T x'$ and that φ_e is currently injective.

(1) If $\varphi_e(t)$ is defined via Action A3, then there must be some $t' =_T t$ such that $\varphi_e(t')$ has already been defined. Furthermore, there is currently some element in $[\varphi_e(t')]_B$ that is not yet in $\mathrm{rng}(\varphi_e)$ and can be chosen to serve as the φ_e image for t. In other words, φ_e still satisfies the desired property and is both primitive recursive and injective.

(2) If $\varphi_e(t)$ is defined via Case 3i of the formal construction, then recall that we search for some $k > 0$ such that $d_e f_e^k r_e(t) \neq_B b$ for any b currently in $\mathrm{rng}(\varphi_e)$. This ensures that φ_e still satisfies the desired properties. It remains to check that this process is primitive recursive. Let n be the current number of elements in $\mathrm{rng}(\varphi_e)$. If the f_e-locus of t does not close; $f_e^i r_e(t) \neq_T r_e(t)$ for each $i < n+1$. Since f_e is assumed to be an isomorphism, then it follows that this gives $n+1$ distinct A-classes, $[r_e(t)]_A, [f_e r_e(t)]_A, \ldots, [f_e^n r_e(t)]_A$. As d_e if also an isomorphism, it follows that $[d_e r_e(t)]_B, [d_e f_e r_e(t)]_B, \ldots, [d_e f_e^n r_e(t)]_B$ are all distinct B-classes. Therefore, at least one of these classes is not currently in $\mathrm{rng}(\varphi_e)$ and can thus act as a φ_e image for t.

Now suppose that the f_e-locus of t closes; there exists some $i < n+1$ such that $f_e^i r_e(t) =_T r_e(t)$. We may further assume that such an i is the least at which this occurs. Similar to before, this implies that $[d_e r_e(t)]_B, [d_e f_e r_e(t)]_B, \ldots, [d_e f_e^i r_e(t)]_B$ are all distinct B-classes. Suppose for a contradiction that for each $j \leq i$, there is some $t_j \in \mathrm{dom}(\varphi_e)$ such that $\varphi_e(t_j) =_B d_e f_e^j r_e(t)$. Now consider the elements $b = d_e r_e(t)$ and $b_j = d_e r_e(t_j)$ for each

$j \leq i$. Since each $t_j \sim_e t$, and the classes $[d_e r_e(t_j)]_B$ are all distinct, then there must be some j such that $d_e r_e(t_j) =_B d_e r_e(t)$. Since $\varphi_e(t)$ is not being defined via Action A3, it implies that $t \neq_T t_j$; one of d_e or r_e cannot be an isomorphism, a contradiction. Therefore, a suitable φ image for t can be found in primitive recursive time via Case 3i. Furthermore, it is clear that the way $\varphi_e(t)$ is defined ensures that it is still an isomorphism.

(3) Finally, we check the case when $\varphi_e(t)$ is defined via Case 3ii of the formal construction. In this case we need not worry about the primitive recursiveness of φ_e as we are only searching for a suitable φ_e preimage for some element b. Recall also that we will define $\varphi_e(t) = b$ via Case 3ii only if such a b satisfies the following conditions.

- $b \neq_B b'$ for any b' currently in $\mathrm{rng}(\varphi_e)$.
- The current size of $[b]_B$ is at least N.
- There are at least sufficiently many T-classes in the f_e-locus of $h(b)$.

However, by the assumption that this version of φ_e is never abandoned, we can then obtain that $\#[b]_B = N$. Furthermore, the T-class that is chosen to map to $[b]_B$ will also never have size $> N$ otherwise we would have abandoned this definition of φ_e.

Therefore, if R_e is attended to for cofinitely many stages, then $\varphi_e : T \to B$ is a surjective primitive recursive isomorphism. □

Since $B <_{pr} T$, by applying the lemma above, it must be that every attempt at defining φ_e must be abandoned at some finite stage. We now show that R_e also cannot begin infinitely many attempts at defining φ_e. Before we state and prove the lemma, we make the following observation.

Fact 3.7. *Let $f : A \to A$ be a primitive recursive isomorphism (possibly non-surjective), then up to a primitive recursive delay d, we have the following.*

- *At each stage s, for each $a \in A_s$, $\#[a]_A^{d(s)} \leq \#[f(a)]_A^s$.*
- *If there exists $k > 0$ such that $f^k(a) =_A a$, then for each $i < k$, and for each $s > k$, $\#[f^i(a)]_A^{d(s)} = \#[a]_A^s$.*

Lemma 3.8. *If R_e swaps infinitely often between being paused and being attended to, then G_e is a N-nice function.*

Proof. We may further assume that e is the least index for which R_e swaps between being paused and being attended to infinitely often. That is to say, past some finite stage, R_e never enters the waiting state. Fix such a stage s, and let the stages $s_0 < s_1 < s_2 < \ldots$ be the infinitely many stages (after s) at which R_e enters the paused state. Whenever R_e enters the paused state at stage s_i, we act as described in Case 3 of the formal construction.

- Define $y_{e,s_i+1} = y_{e,s_i} + 1$.
- For each $s' > s_i$, define $G_{e,s'}(y_{e,s_i})$ to be $\#[h(b^e)]_T^{s'}$. (Recall that b^e is always defined whenever R_e becomes paused.)

That is, $G_e(y) = \lim_s G_{e,s}(y)$ is defined for all values of $y \geq 2$. Furthermore, as T is a primitive recursive presentation of E, it follows from the second point that G_e is a l.m.f.. It then remains to check that E has at least y classes of size $G_e(y)$ for each $y \geq 2$. To do so, we analyse the conditions for R_e to become paused. Recall that in order for a requirement to become paused, one of the following must happen.

(1) There exists some $t \in T_s$ such that $\#[t]^s_T > \#[\varphi_e(t)]^s_B$.

(2) There exists some $b \in B_s$ such that $(**)$ holds.

We consider the two cases below.

Case 1: If $t \in T_s$ is such that $\#[t]^s_T > \#[\varphi_e(t)]^s_B$, then we claim that $\varphi_e(t) =_B d_e f_e^k r_e(t)$ for some $k < 0$, and that the f_e-locus of t has not closed. If the f_e-locus has closed, then we will be able to find some $k' > 0$ such that $\varphi_e(t) =_B d_e f_e^{k'} r_e(t)$ and since d_e, f_e and r_e are all primitive recursive isomorphisms, it cannot be that $\#[t]^s_T > \#[\varphi_e(t)]^s_B$ (recall Fact 3.7).

In order for $\varphi_e(t)$ to be in the same B-class as $d_e f_e^k r_e(t)$ for some $k < 0$, it must be that $\varphi_e(t)$ was defined via Case 3ii of the formal construction. In other words, the b chosen to be the φ_e image of t must satisfy $(*)$; in particular, there are already at least N many elements in $[b]_B$ and there are at least $y_{e,s}$ many distinct T-classes in $[h(b)]_{\sim_e}$. Together with the case assumption, this implies that $b^e = \varphi_e(t)$ satisfies the conditions $(**)$. We may thus assume that we are in Case 2.

Case 2: In order for d_e, f_e, r_e to all be isomorphisms, it must be that all distinct E-classes in $[h(b)]_{\sim_e}$ has the same number of elements. Since $(**)$ implies that there is at least one class in $[h(b)]_{\sim_e}$ that has size $> N$, then all classes in $[h(b)]_{\sim_e}$ should also have sizes $> N$ eventually. In fact, all classes in $[h(b)]_{\sim_e}$ should have the same size eventually. Furthermore, when the requirement is paused at stage s, there are at least $y_{e,s}$ many distinct T-classes in $[h(b)]_{\sim_e}$. Therefore, there will be at least $y_{e,s}$ many distinct T-classes with sizes $G_e(y_{e,s})$.

Thus, as long as d_e, f_e, r_e are all isomorphisms, and R_e is paused and attended to infinitely often, then we must succeed in defining an N-nice G_e. □

Recall that a requirement R_e only ever enters the waiting state if some higher priority requirement leaves the paused state. By the assumptions (premise of Lemma 2.6) that $B <_{pr} T$ and no nice G exists, this allows us to conclude that each requirement R_e is paused for cofinitely many stages. Finally, we show that if this happens, then each R_e is satisfied.

Lemma 3.9. *If R_e is paused for cofinitely many stages, then R_e is satisfied.*

Proof. Fix some stage s such that for all $s' > s$, R_e is in the paused state at stage s'. Recall that in order for a requirement to become paused, one of two things must happen.

(i) There exists some $t \in T_s$ such that $\#[t]^s_T > \#[\varphi_e(t)]^s_B$.

(ii) There exists some $b \in B_s$ such that $(**)$ holds.

If (ii) holds, then there must exist some $t \sim_e h(b)$ such that $\#[t]_T$ currently has size $> N$. We now show that this must also be true if (i) holds. Suppose that (i) holds and fix the t which witnesses (i). Recall that for each $t \in T$, $\varphi_e(t) =_B d_e f_e^k r_e(t)$ for some $k \in \mathbb{Z}$. Since d_e, f_e, r_e are all defined

primitive recursively during the stages s at which R_e is being attended to, then it cannot be that $\#[t]^s_T > \#[d_e f_e^k r_e(t)]^s_B$ for any $k \geq 0$. In other words, it must be that $\varphi_e(t) =_B d_e f_e^k r_e(t)$ for some $k < 0$. Recall however that this only happens when we are attempting to make φ_e surjective and so $b = d_e f_e^k r_e(t)$ satisfies $(*)$ (see Case 3ii of the formal construction). We thus obtain that $\#[t]^s_T > \#[b]^s_B \geq N$ and thus $b = d_e f_e^k r_e(t)$ satisfies $(**)$. Once this is discovered during the construction, b^e must have been defined to be b.

In order for R_e to become ready to be resumed, the f_e-locus of $h(b^e)$ must have closed; thus we may assume that the f_e-locus of $h(b^e)$ never closes. That is, for each $k \geq 0$, $f_e^k r_e h(b^e) \neq_A r_e h(b^e)$. And so, each $k \geq 0$ gives a distinct T-class which must each contain $\#[h(b^e)]_T > N$ elements provided that r_e and f_e are isomorphisms. But T has no size $> N$ which is repeated infinitely often, thus either r_e or f_e cannot be isomorphisms. Therefore R_e is satisfied as either $\beta_{e/2}$ (if e is even) or $\alpha_{(e-1)/2}$ (if e is odd) fails as an isomorphism. □

4. Proof of Lemma 2.7

Recall that Lemma 2.6 allows us to obtain the forward implication of the main theorem. It thus remains to prove the converse which we restate below.

Lemma 2.7. *Let E be an equivalence relation. If there exists nice non-decreasing G_1, G_2 such that for all y, either $G_1(y) < G_2(y)$ or $G_2(y) = \infty$, then the punctual degrees of E is not dense.*

Let E be some equivalence relation such that there are nice non-decreasing functions G_1, G_2 which satisfies the premise. Also fix some effective listing $\{A_i\}_{i\in\omega}$ of primitive recursive equivalence relations, and $\{p_j\}_{j\in\omega}$, $\{q_k\}_{k\in\omega}$, $\{\varphi_e\}_{e\in\omega}$ primitive recursive functions. We aim to construct B, T primitive recursive equivalence relations isomorphic to E, and a primitive recursive isomorphism $h : B \to_{onto} T$, satisfying the following requirements.

$R_{\langle i,j,k\rangle}$: If $p_j : B \to A_i$ and $q_k : A_i \to T$ are primitive recursive isomorphisms, then $\beta_{\langle i,j,k\rangle} : A_i \to B$ or $\alpha_{\langle i,j,k\rangle} : T \to A_i$ is also a primitive recursive isomorphism.

Q_e : $\varphi_e : T \to B$ is not a primitive recursive isomorphism.

As is the usual convention, we replace the indices i, j, k with a single e by applying the pairing function. We shall thus refer to $R_{\langle i,j,k\rangle}, A_i, p_j, q_k$ simply as R_e, A_e, p_e and q_e subsequently.

In contrast to the proof in the previous section, instead of simply 'copying' either the top or bottom structure, we are now responsible for constructing structures B, T which are of the correct isomorphism type. We shall do so by 'copying' some punctual presentation of E. This process will be assumed to take place automatically while the construction happens.

4.1. **Padding.** Fix some primitive recursive presentation of E and let the distinct classes of E be denoted by $[e_0]_E, [e_1]_E, \ldots$. We aim to construct T also with distinct classes $[t_0]_T, [t_1]_T, \ldots$. We split T into two parts, the classes with even indices and the classes with odd indices. Evidently, we cannot copy E exactly. Some level of control over the classes and their sizes should be retained by us in order to satisfy the requirements R_e and Q_e. Thus, we will use the even indexed classes to satisfy the requirements while using the odd indexed classes to ensure that $T \cong E$.

In the construction, each class will have a *tag*. In some sense, the tag will dictate the size of the class. However, the classes will not strictly follow the size of its tag stagewise, as we want to have

the freedom to temporarily stop a class from increasing its size. Instead, we shall ensure that the size of the class in its limit is equal to the size of the tag in the limit. Under this assumption, we define a function f from the classes of T to the classes of E with a stagewise approximation f_s based on the tags of the classes. The details are as follows.

At each stage s, define $f_s([t_i]_T)$ in order of the index i.

- If i is even, let $f_s([t_i]_T) = [e_j]_E$ where j is the least index such that $j \geq i$, $\#[e_j]_E^s$ is currently equal to the tag of $[t_i]_T$, and $[e_j]_E \neq f_s([t_{i'}]_T)$ for any $i' < i$. If no such j currently exists, then we let $f_s([t_i]_T)\uparrow$.
- If i is odd, let $f_s([t_i]_T) = f_{s-1}([t_i]_T)$ if $f_{s-1}([t_i]_T)\downarrow$ and $f_{s-1}([t_i]_T) \neq f_s([t_{i'}]_T)$ for any $i' < i$ and i' even. Otherwise, let $f_s([t_i]_T) = [e_j]_E$ where j is the least index satisfying all of the following.
  - $[e_j]_E \neq f_{s-1}([t_{i'}]_T)$ for any odd i'.
  - $[e_j]_E \neq f_{s-1}([t_{i'}]_T)$ for any even $i' < i$.
  - $\#[e_j]_E^s$ is at least as large as the current tag of $[t_i]_T$.

  If no such j exists, let $f_s([t_i]_T)\uparrow$.

Note that if i is odd, then $f_s([t_i]_T)$ can only be 'displaced' by $f_s([t_{2n}]_T)$ for some $2n < i$. In order to ensure that $\lim_s f_s([t_i]_T)\downarrow$ for all odd i, during stage s of the construction, as long as there is some i odd for which $f_s([t_i]_T)\uparrow$, we do not define $f_s([t_{i'}]_T)$ for any newly enumerated classes with odd index i'. The idea here is that provided that $G_2(y) < \infty$ for all y, then $f_s([t_{2i}]_T)$ eventually stabilises for each i. Since the odd indexed classes can only be 'displaced' by lower indexed even classes, then it follows that $f_s([t_{2i+1}]_T)$ also stabilises. Therefore, $\lim_s f_s$ exist and is an isomorphism from T to E (we prove this formally in Lemma 4.1).

Applying the procedure above, we will produce a presentation of E while maintaining some control over the sizes of the even indexed classes via the functions G_1, G_2, provided that $G_2(y)$ is finite for all y. If we instead have that $G_2(y) = \infty$ for some (and hence cofinitely many) y, then we simply use the odd indexed classes to copy the punctual presentation of E and take $G_1^*(x) = G_2^*(x) = G_2(y)$ for all x as the nice functions. Observe that T as constructed would then consist of infinitely many infinite classes (in all the even indexed classes) joined with E. But since $G_2(y) = \infty$ for some $y \in \omega$ and G_2 is nice and non-decreasing, then E must have infinitely many infinite classes. Thus the presentation of T we produce is in fact still isomorphic to E. With this in mind, we fix the nice functions G_1 and G_2 and proceed with the informal description.

4.2. **Informal description.** The idea for the local strategy would be to start with enumerating a *free* element ($t \in T$ without a preimage under $h : B \to T$) and wait for $\varphi(t)\downarrow$ for each φ. Intuitively, this ensures that there is a 'shift' in φ, more specifically, $\varphi \neq h^{-1}$. Using this fact, we attempt to make φ non-surjective by monitoring whether $h^{-1}(t)$ ever enters the range of φ. If it never does so, then we succeed in satisfying some Q requirement. If it instead enters $\mathrm{rng}(\varphi)$ at some finite stage, then we obtain a diagonalisation by increasing the size of $[\varphi^{-1}h^{-1}(t)]_T$ while keeping $[h^{-1}(t)]_B$ temporarily at its current size. Then φ cannot possibly be a primitive recursive isomorphism if we are able to maintain $\#[h^{-1}(t)]_B < \#[\varphi^{-1}h^{-1}(t)]_T$ for as long as we like. The main tension would thus be to keep $\beta : A \to B$ and $\alpha : T \to A$ primitive recursive while retaining the ability to increase

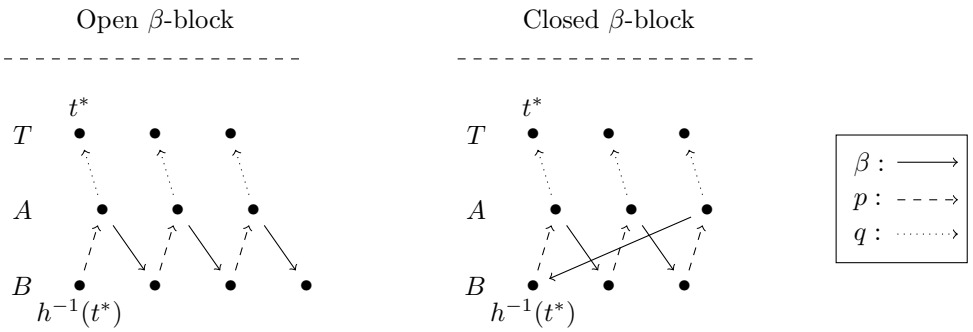


FIGURE 3. The β-block

the size of $[\varphi^{-1}h^{-1}(t)]_T$ when required. To satisfy the requirements, we will construct infinitely many *gadgets*, one at a time, where each gadget satisfies finitely many R and Q requirements.

4.2.1. *Strategy for R.* In order to satisfy R, we need only succeed in defining either $\beta : A \to B$ or $\alpha : T \to A$, which we shall refer to as the Π^0_2 and Σ^0_2 outcome respectively. A single requirement R will have two lists of parameters given by l^β and l^α (to be explained later). For now, we can think of l^β and l^α as dictating the delay β and α are allowed before they have to be defined on newly enumerated elements in their respective structures. Each gadget will be started by enumerating a free class $[t^*]_T$ into T without enumerating its h preimage into B. The idea for the R strategy is to 'guess' whether $q^{-1}(t^*)$ shows up quickly in A. If $q^{-1}(t^*)$ is fast relative to l^α, then we satisfy R via the Σ^0_2 outcome. If $q^{-1}(t^*)$ is slow relative to l^α, then we will satisfy R via the Π^0_2 outcome. Once we have decided which outcome will be used to satisfy R, we enumerate $h^{-1}(t^*)$ into B and say that this gadget is *prepared*.

The Π^0_2 strategy: If $q^{-1}(t^*)$ is 'slow', that is, it is not enumerated into A within the delay allowed by l^α, then we satisfy R via defining β. More specifically, once the time given by l^α runs out, we may then enumerate $h^{-1}(t^*)$ into B and define $\beta q^{-1}(t^*) = h^{-1}(t^*)$. It follows that β is both primitive recursive and surjective on this gadget as it consists only of t^* and its h preimage.

The Σ^0_2 strategy: If $q^{-1}(t^*)$ is 'fast', that is, it is enumerated into A before the time given by l^α runs out, then we are able to define $\alpha(t^*) = q^{-1}(t^*)$. Even though we are satisfying R via defining α, since the Π^0_2 outcome is to the left, then we must keep β defined (but possibly non-surjective) while attempting to define α. Thus, once $q^{-1}(t^*)$ is enumerated into A, we have to enumerate some 'new' class $[b]_B$ into B, for which $h(b) \neq_T t^*$, to serve as the β image for $q^{-1}(t^*)$. Furthermore, in order to keep β primitive recursive and injective, whenever $q^{-1}(t)\downarrow$ for some t in this gadget we must enumerate some appropriate element b to serve as the β image for $q^{-1}(t)$. That is, we may be required to enumerate new classes into B to keep β primitive recursive and injective. On the new elements within these new classes, we then define $\alpha(t) = ph^{-1}(t)$. Since none of these elements are enumerated without a h preimage, then α is guaranteed to be primitive recursive. To ensure that α is an isomorphism, some care is required in ensuring that α is injective, but we leave the details to Section 4.2.3.

Once we enumerate $h^{-1}(t^*)$ (the specifics of when we decide to do so can be found in Sections 4.2.3 and 4.2.4), then we say that the gadget is *prepared*. Assuming that when this happens, and we have chosen the Σ^0_2 outcome for this R requirement, then there are two further possibilities as follows.

(1) We never again visit the Π^0_2 outcome of this R requirement. In this case, the β-block remains open forever (see Fig. 3), then β is not surjective, as $h^{-1}(t^*) \notin \mathrm{rng}(\beta)$. In this case, R is instead satisfied by successfully defining α (to be explained in more detail later).

(2) At some later stage, we again visit the Π^0_2 outcome for R. Then, we make β surjective on this gadget by *closing* the β-block as illustrated in Fig. 3. It becomes clear that β is now surjective on this gadget, but the trade-off is that all classes within the β-block must now be kept at the same size (up to some primitive recursive delay) as $[t^*]_T$ for the sake of β. In other words, we do not want to close the β-block unless we are prepared to sacrifice the ability to keep the classes within the β-block at a different size from the size of $[t^*]_T$.

4.2.2. *Strategy for Q.* As mentioned, when a gadget is started, some free class $[t^*]_T$ is enumerated into T. Let $\varphi : T \to B$ be given and we wait for $\varphi(t^*)\!\downarrow$. If $\varphi(t^*)$ converges and no other classes are enumerated into this gadget, we claim that we are then able to diagonalise φ. As long as we retain the ability to keep distinct gadgets and the portion of the structure dedicated to padding 'sufficiently disjoint', we can increase the size of t^* without worrying about having also to increase the size of $\varphi(t^*)$ at the same rate. (The full diagonalisation procedure is explained in Section 4.3.2.)

Recall that the main idea was to attempt to keep φ non-surjective. In particular, we will never enumerate $h^{-1}(t^*)$ into B until after $\varphi(t^*)\!\downarrow$. Thus, the only possibility that we do not immediately obtain a diagonalisation is if $q^{-1}(t^*)\!\downarrow$ before $\varphi(t^*)$ does. Now according to the strategy described for R above, for the sake of β, we would have to start enumerating new classes b, where $h(b) \neq_T t^*$, to serve as the β image of $q^{-1}(t^*)$. Such a class could then become the φ image for t^*. Observe that if $\varphi(t^*) = (h\beta q^{-1})^k(t^*)$ for some $k > 0$, then in order to keep β primitive recursive, it must be that $\#[t^*]_T \leq \#[\varphi(t^*)]_B$. That is to say, φ cannot be diagonalised without damaging β. The intuition here would then be to use φ to define α (more details on that later).

Once $\varphi(t^*)\!\downarrow = (h\beta q^{-1})^k(t^*)$ for some $k > 0$, then enumerate $b^* \coloneqq h^{-1}(t^*)$ into B, keep the β-block in this gadget open, and wait for $\varphi^{-1}(b^*)\!\downarrow$.

- If $\varphi^{-1}(b^*)\!\downarrow$ and is not in the same gadget as b^*, then close the β-block, and wait for φ to converge on all classes $t \in T$ currently within this gadget. By pigeonhole principle, since the number of classes in B and T within this gadget are the same, there must exist some $t \in T$ for which $\varphi(t)$ is not within this gadget. Once again, we may then diagonalise φ as long as the different parts of the structure are 'sufficiently disjoint'. (Again we refer the reader to Section 4.3.2 for the full diagonalisation procedures.)

- If $\varphi^{-1}(b^*)$ is in the same gadget as b^*, then note that the β-block is still open. Therefore, we are able to increase the size of $[\varphi^{-1}(b^*)]_T$ (which must be from the β-block) while temporarily preventing $[b^*]_B$ and $[t^*]_T$ from increasing in size. We illustrate this further using Fig. 4. Since we are only committed to maintaining the primitive recursiveness of β, when increasing the size of some class at the 'back' of the gadget, there is no reason for us to increase the size of $[b^*]_B$. As long as the gadget is not closed, when increasing the size of some class $[b]_B$, we need only increase the sizes of 'subsequent' classes given by $[(\beta q^{-1}h)^n(b)]_B$ for $n > 0$. Then we need only wait for some finite amount of time for φ to

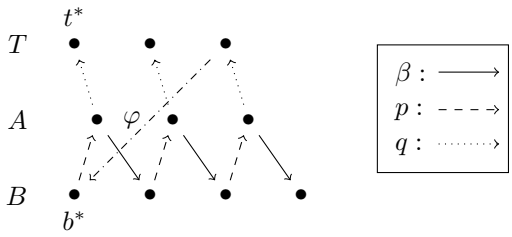
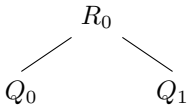


FIGURE 4. Diagonalising φ

converge on the new elements introduced to $[\varphi^{-1}(b^*)]_T$ to obtain a diagonalisation, thus satisfying Q. Once φ is observed to fail, we close the β-block.

Throughout the construction, each gadget will be used at most once to obtain a diagonalisation against some φ as described above.

4.2.3. *One R strategy and two Q strategies.* We arrange the strategies on a tree. Let us assume for now that we only have R_0, Q_0, Q_1 arranged as follows.



Let $l_0^\beta = \emptyset$ and $l_0^\alpha = \{\varphi_0\}$, where φ_0 is given by Q_0. Recall that these lists dictate the delay allowed by β_0 and α_0 respectively. More specifically, in this setup, β_0 is not allowed any delay, while α_0 need not be defined until $\varphi_0\!\downarrow$.

As before, we begin a gadget by enumerating a free class $[t^*]_T$ into T. We enumerate no other classes while waiting for either $\varphi_0(t^*)\!\downarrow$ or $q_0^{-1}(t^*)\!\downarrow$. We may assume that $q_0^{-1}(t^*)$ converges before $\varphi_0(t^*)$, otherwise we simply enumerate $h^{-1}(t^*)$ into B and declare the gadget prepared (as $\varphi_0(t^*)$ converges to some class outside of this gadget and will be diagonalised). Once $q_0^{-1}(t^*)\!\downarrow$, to maintain the primitive recursiveness of β_0, we have to start enumerating classes to form the β_0-block. We may further assume that $\varphi_0(t^*)$ must lie within this gadget, and hence must be one of the classes in the β_0-block. Roughly speaking, this means that φ_0 is 'copying' β_0, and since β_0 is associated with the branch above Q_0, we cannot satisfy φ_0 while successfully defining β_0. Furthermore, since the delay allowed by $l_0^\alpha = \{\varphi_0\}$ is over, we must start the definition of α_0 now.

$$\alpha_0(t) = \begin{cases} q_0^{-1}(t) & \text{if } t =_T t^*, \\ p_0 h^{-1}(t) & \text{otherwise.} \end{cases}$$

While we are preparing the gadget we keep all classes within the gadget temporarily at size 1. As mentioned previously, if $\varphi_0(t^*)\!\downarrow$ and $q_0^{-1}(t^*)\!\uparrow$, then we are able to diagonalise against φ_0. Thus, at least on t^*, α_0 is primitive recursive. On all other classes, $\alpha_0 = p_0 h^{-1}$, which is always primitive recursive as h^{-1} is possibly only slow on t^*. It is also not too hard to see that before the gadget is prepared, $q_0^{-1}(t^*) \notin \mathrm{rng}(p_0)$, otherwise we are able to diagonalise against $q_0 p_0$ by increasing the sizes of the classes at the 'back' of the gadget and temporarily keeping $\#[t^*]_T = 1$. (For the formal and more general procedure, see Section 4.3.2.)

Now that we have decided to pursue the Σ^0_2 outcome for R_0, we wait for $\varphi_1(t^*)\downarrow$. As before, if $\varphi_1(t^*)$ does not lie within this gadget, then we will diagonalise against φ_1. Thus, we may assume that $\varphi_1(t^*)$ is within the β_0-block. The intuition here is that φ_1 can only 'copy' β_0, as this is the only block within this gadget. Once this is observed, enumerate $h^{-1}(t^*)$ into the gadget and we say that the gadget is prepared. The main difference between φ_0 and φ_1 'copying' β_0 is that we are allowed to keep β_0 not surjective for the satisfaction of Q_1, but not for the satisfaction of Q_0. Thus, we only enumerate $h^{-1}(t^*)$ after we find that $\varphi_1(t^*) = (h\beta_0 q_0^{-1})^k(t^*)$ for some $k > 0$.

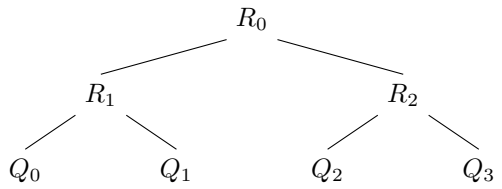
Once the gadget is prepared, we will tag (recall from Section 4.1) each class with $G_1(y)$ for some appropriate y and let the classes grow. This potentially causes issues in maintaining the primitive recursiveness of α_0 on new elements $t =_T t^*$. To keep α_0 primitive recursive while growing the individual classes, we ensure that $q_0^{-1}(t^*)$ enters $\mathrm{rng}(p_0)$ after the gadget becomes prepared. For q_0, p_0, we ensure that $q_0^{-1}(t^*) =_{A_0} p_0 h^{-1}(t^*)$ and keep $\#[h^{-1}(t^*)]_B = \#[t^*]_T$ for every subsequent stage. If such a condition is not met by q_0, p_0 after the gadget is prepared, then we will be able to diagonalise against q_0 or p_0 in finite time, thus satisfying R_0 (more details in Section 4.3.2).

On this gadget, we can thus successfully define α_0 while keeping φ_1 not surjective. In this scenario, we will say to *assign* Q_1 to this gadget. As long as Q_1 is assigned to this gadget, we never close the β_0-block. At present, the status of each requirement is as follows.

- R_0 is temporarily satisfied by α_0 being primitive recursive and surjective (as long as p_0 is also surjective). β_0 though not currently surjective, is still kept primitive recursive.
- Q_0 is currently to the left of the path of the construction and thus we may ignore its status. But in the event that α_0 fails to be defined for some $t \in [t^*]_T$, we have a guarantee that φ_0 also fails as a primitive recursive isomorphism and thus can return to pursuing the Π^0_2 outcome for R_0.
- Q_1 is temporarily satisfied as $h^{-1}(t^*) \notin \mathrm{rng}(\varphi_1)$. If $h^{-1}(t^*)$ enters the range of φ_1, then as discussed in Section 4.2.2, we may obtain a diagonalisation against φ_1 since the β_0-block is still open. Once the diagonalisation is obtained, we may then close the β_0-block and note that this would place $h^{-1}(t^*)$ into the range of β_0, thus making β_0 surjective on this gadget. Furthermore, if we have yet to find a diagonalisation against φ_0 on this gadget, then α_0 will also still be a primitive recursive surjective isomorphism.

In summary, after Q_1 is assigned to the gadget and it is prepared, we either wait forever for $h^{-1}(t^*)$ to enter $\mathrm{rng}(\varphi_1)$, or find some finite stage at which we successfully witness the failure of φ_1. On the other hand, if on some later gadget, we decide to pursue the Π^0_2 outcome for R_0, then we close the β_0-block in this gadget to place $h^{-1}(t^*)$ into $\mathrm{rng}(\beta_0)$, making the gadget consistent with the construction.

4.2.4. *Three R strategies and multiple Q strategies.* We once again place the different requirements on a tree given as follows.



When there are multiple R requirements, the gadget could now potentially consist of multiple β-blocks. Recall that when $q_i^{-1}(t^*)\downarrow$, we enumerate the β_i-block (if the gadget is currently not prepared). Therefore, when there are multiple R requirements, $q_i^{-1}(t^*)$ could converge for multiple i before we are ready to enumerate $h^{-1}(t^*)$. As a result, other than defining β_i on the β_i-block, we also need to keep β_i primitive recursive on the β_j-blocks for $j \neq i$. This can easily be done by taking $\beta_i(t) = h^{-1}q_i(t)$ for each $t \neq_T t^*$ and t not within the β_i-block. There is no issue with taking this definition as h^{-1} is only ever 'slow' on the free class $[t^*]_T$. Thus, $\beta_i(t)$ will always converge in primitive recursive time once $q_i(t)\downarrow$. In addition, it is easy to see that β_i is still only ever non-surjective if the β_i-block is enumerated and left open.

Before we proceed to the strategy to satisfy the various requirements, we compute the parameters for each β_i and α_i. In general, β_i is allowed up to a delay of all the φ_j functions given by the Q_j which are lexicographically left of R_i. In addition to these, α_i is also allowed to wait for those φ_j which are descendants of the left branch of R_i. Applying this procedure allows us to obtain the following.

- $l_0^\beta = \emptyset$, $l_0^\alpha = \{\varphi_0, \varphi_1\}$.
- $l_1^\beta = \emptyset$, $l_1^\alpha = \{\varphi_0\}$.
- $l_2^\beta = \{\varphi_0, \varphi_1\}$, $l_2^\alpha = \{\varphi_0, \varphi_1, \varphi_2\}$.

As before, begin the gadget by enumerating a free class $[t^*]_T$ into T and compute $\varphi_i(t^*)$ in order of the index i. We may once again assume that each $\varphi_i(t^*)$ converges to some class within the gadget, otherwise we simply enumerate $h^{-1}(t^*)$ into B and begin the diagonalisation procedure to satisfy Q_i. While waiting for $\varphi_0(t^*)\downarrow$, one of $q_j^{-1}(t^*)$ could converge for $j = 0, 1, 2$. Recall that if $q_j^{-1}(t^*)$ converges, we start a β_j-block in order to maintain the primitive recursiveness of β_j. However, since β_2 is allowed to wait for φ_0, φ_1 to converge on t^* first, even if $q_2^{-1}(t^*)$ is enumerated into A_2, we do not begin building the β_2-block. We note here that there is no real disadvantage even if we were to begin the β_2-block; it just leads to more cases when considering the φ_0 image of t^*. As such, we turn our focus to $q_0^{-1}(t^*)$ and $q_1^{-1}(t^*)$.

If $q_1^{-1}(t^*)$ is enumerated 'quickly' into A_1, then the strategy to satisfy R_1 was to attempt to define α_1 surjectively; $\alpha_1(t^*) = q_1^{-1}(t^*)\downarrow$. However, if we choose to do so, we need the ability to keep α_1 primitive recursive even after the gadget is prepared. To this end, there will be two main mechanisms for ensuring the above.

(M1) The first way is to ensure that $q_1^{-1}(t^*) =_{A_1} p_1(b)$ for some b where $[b]_B$ can be kept at the same size as $[t^*]_T$. In particular, for q_1, p_1, such a b must be contained within the β_0-block. More generally, for q_i, p_i, the b for which $q_i^{-1}(t^*) =_{A_i} p_i(b)$ must be contained in some block associated with R_j for some $j < i$, or we will be able to diagonalise against q_i or p_i (see Section 4.3.2). Which means, if we decide to define $\alpha_1(t^*) = q_1^{-1}(t^*)$, then we need to commit to closing the β_0-block once the gadget becomes prepared.

(M2) The second way is via *challenging* some $\varphi \in l_1^\alpha$. To illustrate this, suppose that $\varphi_0(t^*)\downarrow$ is contained in the β_1-block. That is, $\varphi_0(t^*) = \beta_1 q_1^{-1}(h\beta_1 q_1^{-1})^k(t^*)$ for some $k \geq 0$. We are then able to 'force' $\#[q_1^{-1}(t^*)]_{A_1}$ to increase by keeping all classes within the β_1-block the same size as $[q_1^{-1}(t^*)]_{A_1}$. This ensures that as long as $[q_1^{-1}(t^*)]_{A_1}$ does not grow, then $\#[\varphi_0(t^*)]_B \leq \#[q_1^{-1}(t^*)]_{A_1} < \#[t^*]_T$, allowing us to obtain a diagonalisation against φ_0.

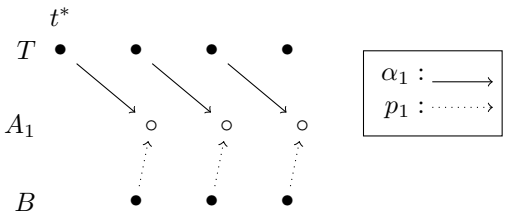


FIGURE 5. The α_1-block

While a gadget is being prepared, we do not know which blocks will end up becoming closed as that depends on the behaviour of the various φ_i. But α_1 needs to be defined within the delay allowed by l_1^α; we cannot wait until the gadget is prepared before deciding whether or not to define $\alpha_1 = q_1^{-1}$. To ensure that we are able to keep α_1 primitive recursive after the gadget is prepared regardless of which blocks become closed, we can only define $\alpha_1(t^*) = q_1^{-1}(t^*)$ if we are able to guarantee that at least one of the two mechanisms work; in particular, we will define $\alpha_1(t^*) = q_1^{-1}(t^*)$ only if $\varphi_i(t^*)$ is in the β_1-block for some $\varphi_i \in l_1^\alpha$. Similarly, we will pursue the Σ_2^0 outcome for R_0 only if $\varphi_i(t^*)$ is in the β_0-block for some $\varphi_i \in l_0^\alpha$. We consider some the different possibilities for the position of $\varphi_0(t^*)$ below.

(1) If $\varphi_0(t^*)$ is in the β_0-block, then we pursue the Σ_2^0 outcome for R_0 as mentioned previously. This would however mean that both the Π_2^0 and Σ_2^0 outcomes for R_1 are now lexicographically left of the current path of our construction. Therefore, we need to keep $\alpha_1(t^*)$ defined using only the delay allowed by φ_0. To do so, we begin a new block which we shall call the α_1-block. Enumerate classes $[b]_B$ such that $h(b) \not\equiv_T t^*$ to generate new classes $[p_1(b)]_{A_1}$ in A_1 that keep α_1 primitive recursive (see Fig. 5). Evidently, as long as the α_1-block remains open, α_1 will not be surjective as $p_1 h^{-1}(t^*) \notin \mathrm{rng}(\alpha_1)$. But just as before, if we ever visit the Σ_2^0 outcome for R_1 (or some other strategy lexicographically left), then we will close the α_1-block so as to place $p_1 h^{-1}(t^*)$ into its range.

(2) If $\varphi_0(t^*)$ is in the β_1-block, then we pursue the Σ_2^0 outcome for R_1 and proceed to computing $\varphi_1(t^*)$. Note here that even though $q_0^{-1}(t^*)$ may have converged by the time $\varphi_0(t^*)$ converges, we do not commit to the definition of $\alpha_0(t^*) = q_0^{-1}(t^*)$ at this point as we are unable to ensure that one of (M1) or (M2) will hold for α_0. In contrast to the previous case, there is no need to begin a α_0-block here even though $\varphi_0(t^*)$ is not in the β_0-block. The difference is that $l_0^\alpha = \{\varphi_0, \varphi_1\}$ whereas $l_1^\alpha = \{\varphi_0\}$. In other words, $\alpha_0(t^*)$ need not be defined until after $\varphi_1(t^*)\!\downarrow$.

To summarise, below are some of the key ideas behind the actions taken for each strategy.

- Whenever $q_i^{-1}(t^*)\!\downarrow$ for some i, if $\varphi_j(t^*)$ has already converged for each $\varphi_j \in l_i^\beta$, then we enumerate the β_i-block into the gadget. As a result of this, β_i would temporarily be non-surjective within this gadget.
- If $\varphi_j(t^*)\!\downarrow$ and is in the β_i-block where $\varphi_j \in l_i^\alpha$, then we define $\alpha_i(t^*) = q_i^{-1}(t^*)$.
- If $\varphi_j(t^*)\!\downarrow$ for every $\varphi_j \in l_i^\alpha$ but none of $\varphi_j(t^*)$ is in the β_i-block, then we enumerate the α_i-block into the gadget, keeping α_i primitive recursive with the parameters in l_i^α. We will do so regardless of whether $q_i^{-1}(t^*)$ has converged or not.

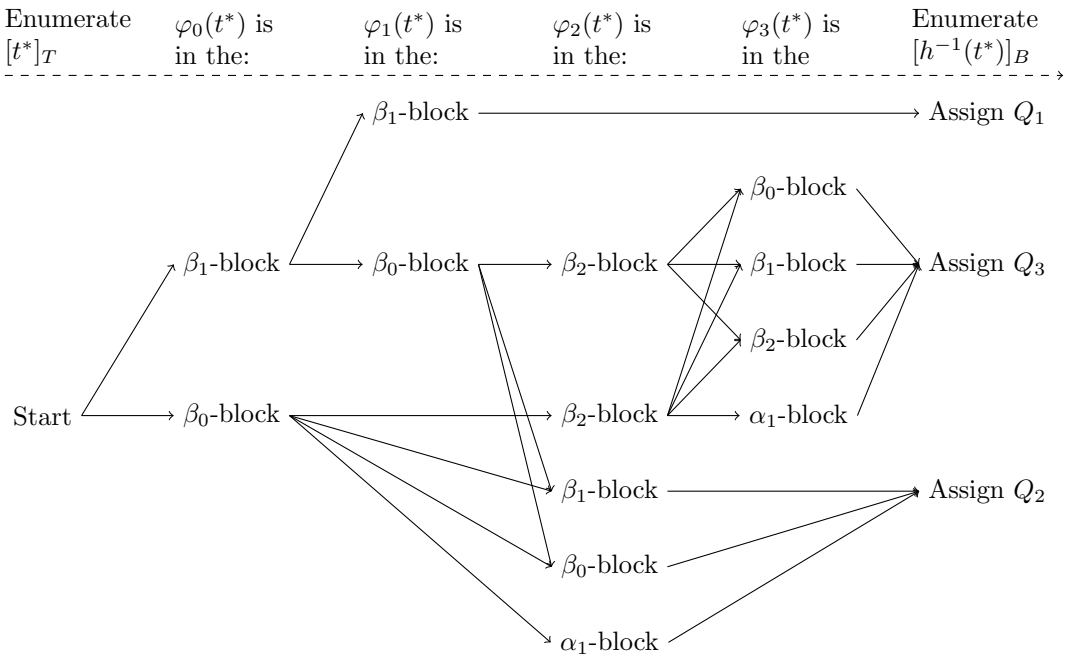


FIGURE 6. Procedure for a single gadget. Note that if $\varphi_0(t^*)$ is in the β_1-block, no α_1-block will be enumerated.

- Once some $\varphi_i(t^*)$ has converged to some element in a block associated with some strategy lexicographically left of φ_i, we will assign Q_i to the gadget and enumerate $[h^{-1}(t^*)]_B$. The idea here is that this gadget is a witnessof the non-surjectivity of φ_i.

Fig. 6 illustrates the procedure for a gadget aiming to satisfy the tree of requirements presented at the beginning of Section 4.2.4. Start by enumerating the free class $[t^*]_T$ and compute $\varphi_i(t^*)$ for the various i in lexicographic order. Once some $\varphi_i(t^*)$ converges to some block associated with an outcome lexicographically left of Q_i, then we assign Q_i to the gadget, enumerate $[h^{-1}(t^*)]_B$, and close all blocks within this gadget except the one $\varphi_i(t^*)$ is contained in.

Note that Q_0 will never be assigned as it is the left-most Q requirement. Intuitively, a Q requirement being assigned to the gadget means that φ is being kept non-surjective at the cost of keeping some other map either β_e or α_e non-surjective. This only makes sense if β_e or α_e is associated with some outcome lexicographically left of Q, and thus Q_0 will never be assigned as there are no outcomes lexicographically left of it.

It is not too hard to see that by the time $\varphi_3(t^*)\downarrow$, at least one of the $\varphi_i(t^*)$ where $i \leq 3$ must lie within some block which is associated with an outcome that is lexicographically left of Q_i, allowing us to finish preparing this gadget. Furthermore, the various statuses of the maps are consistent with the satisfaction of the Q_i being assigned to this gadget (see the end of Section 4.2.5 for a more detailed discussion).

4.2.5. *After a gadget is prepared.* Recall that while a gadget is being prepared, all classes are temporarily left at size 1. Once it is prepared, we need to begin tagging the classes with some target size so as to produce a copy of E in the limit. The tags will be based on the functions G_1, G_2 and the specifics can be found in Section 4.4. The main issue while attempting to grow the classes would be in keeping the various α_e primitive recursive while being consistent with the strategy to satisfy the assigned Q requirement. As mentioned previously, there are two main mechanisms we shall use to force $[q_e^{-1}(t^*)]_{A_e}$ to grow when $[t^*]_T$ increases in size. We provide some details below.

When a gadget becomes prepared, there will only be one block which is left open. Suppose that this block is associated with either the Π^0_2 or Σ^0_2 strategy for R_n. For each α_e where $e \leq n$ and $\alpha_e(t^*) = q_e^{-1}(t^*)$, we will use (M1) to keep α_e primitive recursive. Recall that this means that we have to ensure $q_e^{-1}(t^*) =_{A_e} p_e(b)$ for some b contained in a block associated with some strategy for a higher priority $R_{e'}$ ($e' < e$) or for $b = h^{-1}(t^*)$. Since all but one block is closed, it follows that every block associated with some strategy for $R_{e'}$ where $e' < e \leq n$ must also be closed. That is to say, there are only finitely many distinct classes in B within these blocks, say $[b_0]_B, [b_1]_B, \ldots, [b_m]_B$. If $q_e p_e h^{-1}(t^*), q_e p_e(b_0), q_e p_e(b_1), \ldots, q_e p_e(b_m)$, are all not contained in $[t^*]_T$, then by pigeonhole principle, there is some $b \in \{h^{-1}(t^*), b_0, b_1, \ldots, b_m\}$, such that $q_e p_e(b)$ is contained in some block associated with some strategy for $R_{e''}$ where $e'' \geq e$. Therefore, we are able to temporarily stop the size of $[q_e p_e(b)]_T$ from increasing whilst increasing the size of $[b]_B$, thus obtaining a diagonalisation against $q_e p_e$ (see Section 4.3.2 for more details). Thus, once a gadget becomes prepared, for $e \leq n$, $q_e^{-1}(t^*) =_{A_e} p_e(b)$ either for $b = h^{-1}(t^*)$ or for some b contained in some block associated with some strategy for a higher priority requirement. As long as we commit to keeping the classes contained within all blocks of higher priority than n at the same size as $[t^*]_T$, we are able to keep α_e primitive recursive on $t \in [t^*]_T$.

For α_e where $e > n$ and $\alpha_e(t^*) = q_e^{-1}(t^*)$, we keep them primitive recursive via (M2); by challenging some $\varphi \in l^e_\alpha$. Recall that this means keeping the classes within the β_e-block at the same size as $[q_e^{-1}(t^*)]_{A_e}$. As long as $\#[q_e^{-1}(t^*)]_{A_e} < \#[t^*]_T$, all classes in the β_e-block remains at a size strictly smaller than $\#[t^*]_T$. Thus, we obtain that $\#[t^*]_T > \#[q_e^{-1}(t^*)]_{A_e} = \#[\varphi(t^*)]_B$ for the $\varphi \in l^\alpha_e$ where $\varphi(t^*)$ is in the β_e-block. Such a φ must exist, otherwise we would not have chosen to define $\alpha_e(t^*) = q_e^{-1}(t^*)$. Thus, either $[q_e^{-1}(t^*)]_{A_e}$ grows to the size of $[t^*]_T$, allowing us to keep α_e primitive recursive, or we obtain a diagonalisation against some $\varphi \in l^\alpha_e$ (necessarily lexicographically left of the Σ^0_2 outcome for R_e). Note here that we only need to keep the classes within the β_e-block the same size as $[q_e^{-1}(t^*)]_{A_\sigma}$ for as long as we are defining $\alpha_e(t^*) = q_e^{-1}(t^*)$. Once some $\varphi \in l^\alpha_e$ is discovered to fail, we can (and will) initialise this definition of α_e and allow the classes within the β_e-blocks to grow and match their tags. This will be important in ensuring that the structure we construct is isomorphic to E in the limit.

Finally, for α_e-blocks where $e > n$, we simply keep the classes within them at the same size as $[t^*]_T$. It is easy to see that this is consistent with the strategies described thus far and serves to keep the various α_e where $\alpha_e(t^*) \neq q_e^{-1}(t^*)$ primitive recursive. In summary, the classes within all blocks associated with R_e where $e \leq n$ need to be kept at the same size as $[t^*]_T$ while blocks associated with R_e where $e > n$ must always have sizes $\leq \#[t^*]_T$.

We now explain how the conditions above may be maintained while satisfying the Q requirement assigned to this gadget. Suppose that Q_i was assigned to the gadget; $\varphi_i(t^*)$ is contained in either the β_n-block or the α_n-block. Recall that the basic idea to satisfy Q_i is to wait for $\varphi_i^{-1}h^{-1}(t^*)\downarrow$ and keep $\#[\varphi_i^{-1}h^{-1}(t^*)]_T > \#[h^{-1}(t^*)]_B$ until φ_i fails. Observe that if $\varphi_i^{-1}h^{-1}(t^*) = h(b)$ for some

b in a block associated with R_e where $e \neq n$, then we are unable to keep $\#[h(b)]_T > \#[h^{-1}(t^*)]_B$ without damaging the primitive recursiveness of some α or β. To circumvent this, rather than simply waiting for $h^{-1}(t^*)$ to enter $\mathrm{rng}(\varphi_i)$, we wait for $\mathbf{b} \subseteq \mathrm{rng}(\varphi_i)$ where $\mathbf{b}$ contains $h^{-1}(t^*)$ and a single representative from each class within the closed blocks. Clearly, $\mathbf{b}$ contains only finitely many elements, and if $\mathbf{b} \nsubseteq \mathrm{rng}(\varphi_i)$, then φ_i is not surjective. On the other hand, once $\mathbf{b} \subseteq \mathrm{rng}(\varphi_i)$, it must be that $\varphi_i^{-1}(b) =_T h(b')$ for some $b \in \mathbf{b}$ and b' in the open block. To see why this is so, let $\mathbf{b} = \{h^{-1}(t^*), b_0, b_1, \ldots, b_m\}$ and suppose for a contradiction that for each $b \in \mathbf{b}$, there exists $b' \in \mathbf{b}$ such that $\varphi_i^{-1}(b) =_T h(b')$. Since $\varphi_i(t^*)$ maps to some element in the open block, we know that $\varphi_i(t^*) \neq_B b$ for any $b \in \mathbf{b}$. Thus, by pigeonhole principle, there exists two distinct elements $c, d \in \mathbf{b}$ such that $\varphi_i^{-1}(c) =_T \varphi_i^{-1}(d)$, meaning that φ_i is not an isomorphism. Therefore, once $\mathbf{b} \subseteq \mathrm{rng}(\varphi_i)$, we either discover that φ_i is not an isomorphism, or that there is some $b \in \mathbf{b}$ such that $\varphi_i^{-1}(b) \neq_T h(b')$ for any $b' \in \mathbf{b}$. In the latter case, we will diagonalise against φ_i by increasing the sizes of all classes within the open block while temporarily freezing the sizes of those in the closed blocks (see Section 4.3.2). In this way, we satisfy Q_i while keeping the various α primitive recursive. Once φ_i is witnessed to fail, we will close all blocks within the gadget and never return to it again.

Placing the strategies together, we now provide a brief explanation of how each requirement is satisfied. We refer the reader again to the tree at the beginning of Section 4.2.4 and the flowchart in Fig. 6.

Q_0 **is satisfied:** Note that Q_0 will never be assigned to the gadget since it is the left-most outcome. More specifically, the only way we will finish preparing the gadget while satisfying Q_0 is if $q_i^{-1}(t^*)$ are all 'slow' and do not converge before $\varphi_0(t^*)$. If this happens, then $\varphi_0(t^*)$ must be in some other gadget and we can thus obtain a diagonalisation against it.

In addition, since $q_0^{-1}(t^*)$ and $q_1^{-1}(t^*)$ both do not converge until after $\varphi_0(t^*)\!\downarrow$, then we can define both β_0, β_1 surjectively on this gadget by letting $\beta_0 q_0^{-1}(t^*) = h^{-1}(t^*)$ and $\beta_1 q_1^{-1}(t^*) = h^{-1}(t^*)$. That is, this gadget becomes prepared with only one class in B and one class in T.

Q_1 **is assigned:** Following the flowchart in Fig. 6, the only time Q_1 is assigned to the gadget once it's prepared is if both $\varphi_0(t^*)$ and $\varphi_1(t^*)$ are in the β_1-block. That is to say, $q_1^{-1}(t^*)$ must have converged before $\varphi_0(t^*)$, and thus, $\alpha_1(t^*) = q_1^{-1}(t^*)$ is defined quickly relative to $l_1^\alpha = \{\varphi_0\}$.

Once the gadget becomes prepared, since $\varphi_1(t^*)$ is within the β_1-block, all other blocks must have become closed once the gadget is prepared. Thus, even if $q_0^{-1}(t^*)\!\downarrow$ before $h^{-1}(t^*)$ is enumerated, β_0 is surjective on this gadget as the β_0-block is closed once the gadget is prepared. In addition, α_1 remains primitive recursive, particularly on $[t^*]_T$, as $q_1^{-1}(t^*) =_{A_1} p_1(b)$ for some b in the (closed) β_0-block or $b = h^{-1}(t^*)$. And so, R_0 is satisfied via its Π_2^0 strategy, R_1 is satisfied via its Σ_2^0 strategy, and Q_1 is satisfied because once φ_1 attempts to become surjective on this gadget, it will be diagonalised.

Q_2 **is assigned:** In order for Q_2 to be assigned to the gadget, observe that one of $\varphi_0(t^*)$ or $\varphi_1(t^*)$ must be within the β_0-block. Thus, we are able to define α_0 surjectively on this gadget by taking $\alpha_0(t^*) = q_0^{-1}(t^*)$ within the delay allowed by its parameters $l_0^\alpha = \{\varphi_0, \varphi_1\}$. Furthermore, once the gadget becomes prepared, since R_0 is the highest priority requirement, $q_0^{-1}(t^*) = p_0 h^{-1}(t^*)$, thus keeping α_0 primitive recursive.

Turning our attention to R_1, which is lexicographically left of Q_2, if $\varphi_0(t^*)$ is not within the β_1-block, either because $q_1^{-1}(t^*)$ is 'slow' or simply because $\varphi_0(t^*)$ is in the β_0-block instead, then we would have started the α_1-block, thus keeping $\alpha_1(t^*)$ primitive recursive (with the parameters l_1^α). On the other hand, if $\varphi_0(t^*)$ is in the β_1-block, then we would have defined $\alpha_1(t^*) = q_1^{-1}(t^*)$. Once the gadget is prepared, if the β_0-block becomes closed, then α_1 can be kept primitive recursive for the same reasons as before. However, there is no guarantee that this will happen. In fact, if $\varphi_2(t^*)$ is in the β_0-block, then it must remain open to satisfy Q_2. In this case, we will keep α_1 primitive recursive via challenging $\varphi_0 \in l_1^\alpha$. Therefore, α_1 can be kept primitive recursive regardless of where $\varphi_0(t^*)$ is, and thus both the Π_2^0 and Σ_2^0 outcomes of R_1 can be preserved.

Finally, R_2 will be satisfied via its Π_2^0 outcome. If $q_2^{-1}(t^*)$ is 'slow' ($q_2^{-1}(t^*)\uparrow$ until after $h^{-1}(t^*)$ is enumerated), then we define $\beta_2 q_2^{-1}(t) = h^{-1}(t)$ for any t within this gadget. Note that regardless of whether $q_2^{-1}(t^*)$ is 'fast' or 'slow', we would define $\beta_2 q_2^{-1}(t) = h^{-1}(t)$ for any $t \neq_T t^*$ and not within the β_2-block. If we instead have that $q_2^{-1}(t^*)$ converges before the gadget is prepared, we would have enumerated the β_2-block to keep β_2 primitive recursive, but temporarily non-surjective. Since Q_2 is assumed to be assigned to this gadget, then it cannot be that $\varphi_2(t^*)$ is within the β_2-block (see Fig. 6), and thus once the gadget is prepared, the β_2-block becomes closed, making β_2 surjective.

In summary, Q_2 is satisfied as φ_2 cannot become surjective (or it gets diagonalised), R_0 is satisfied as $q_0^{-1}(t^*)$ converges before $\varphi_0(t^*)\downarrow$ or $\varphi_1(t^*)\downarrow$, R_1 is preserved, and R_2 is satisfied via defining β_2 surjectively.

Q_3 is assigned: The arguments for why R_0 is satisfied via its Σ_2^0 outcome and why both outcomes of R_1 are preserved is similar to the previous cases. In order for Q_3 to become assigned to the gadget, $\varphi_2(t^*)$ must be within the β_2-block; $q_2^{-1}(t^*)$ converged within the delay allowed by l_2^α, and α_2 was defined such that $\alpha_2(t^*) = q_2^{-1}(t^*)$. Once the gadget becomes prepared, if $\varphi_3(t^*)$ is in the β_2-block, then α_2 will be kept primitive recursive via (M1) as all blocks associated with higher priority outcomes are closed. If $\varphi_3(t^*)$ is instead in some block associated with a higher priority outcome, then α_2 is kept primitive recursive via challenging $\varphi_2 \in l_2^\alpha$. Note that regardless of the block which $\varphi_3(t^*)$ is contained in, it must be associated with some outcome lexicographically left of Q_3. Thus, leaving it open and consequentially making the associated map non-surjective is consistent with the current path of the construction. Therefore, we successfully define α_0 and α_2 surjectively, and keep φ_3 non-surjective.

4.2.6. *A global overview of the construction.* We arrange the strategies on a tree. The details can be found in Section 4.3. At each stage of the construction, we will explore a finite subtree and construct a gadget that aims to satisfy one Q requirement. Each gadget considers some finite cover of all possible paths through the tree and becomes prepared by assigning some Q requirement to the gadget. Recall that the intuition behind assigning a Q requirement is that this gadget will witness the non-surjectivity of φ given by the assigned Q requirement. We provide some more details below.

Let $\{\varphi_0, \varphi_1, \ldots, \varphi_k\}$ be a list in lexicographic order of all current terminal nodes in the finite subtree we are considering. As mentioned, we compute $\varphi_i(t^*)$ for each i in lexicographic order. Once some $\varphi_i(t^*)$ is found to be contained within a block associated with a strategy lexicographically left (on the tree), we will assign φ_i to the gadget, enumerate b^* and declare it prepared. At the next stage,

we will extend the finite subtree by adding the descendants of the node associated with φ_i. (See Section 4.4 for more details.)

In Lemma 4.3, we shall show that the procedure above produces finite subtrees with depths that tend to infinity. The true path of the construction will be the left-most path in the union of all the finite subtrees.

4.3. **The tree of strategies.** We arrange the strategies on a subtree $\Gamma \subseteq \{\infty, f\}^{<\omega}$ where $\sigma \in \Gamma$ iff $\sigma(2n+1) = \infty$ for all $n < \frac{|\sigma|}{2}$. We induce a lexicographic ordering on nodes of the tree by letting $\infty \prec f$. For strings σ and τ, we write $\sigma \subseteq \tau$ to mean σ is a prefix of τ, and we use σ^- to denote ξ such that $\xi^\frown x = \sigma$ for some x. Each node represents some strategy as follows.

- The various versions of Q_e are represented by the nodes σ where $|\sigma| = 2e + 1$.
- The various versions of R_e are represented by the nodes σ where $|\sigma| = 2e$.

In view of this, we omit the index of the requirements and instead refer to them directly as R_σ or Q_σ respectively. The requirements will be arranged with the usual priority; σ is of higher priority than τ if $\sigma \prec \tau$ or $\sigma \subsetneq \tau$.

During the construction, we maintain a list Γ_n consisting of all the nodes that have been visited so far up to the n^{th} gadget. Each requirement R_σ also has two lists of parameters given by the following.

- l^β_σ containing all $\tau \in \Gamma_n$ terminal and $\tau \prec \sigma$.
- l^α_σ containing all $\tau \in \Gamma_n$ terminal and $\tau \prec \sigma^\frown f$.

Recall that the lists l^β_σ and l^α_σ dictate the delay which is allowed by the current versions of β_σ and α_σ respectively.

4.3.1. *The gadgets.* Within the n^{th} gadget, we use $[t^n_0]_T$ to denote the free class enumerated at the start of the n^{th} gadget. In addition, for the requirement R_σ, where $|\sigma| = 2e$ for some e, we reserve the $\sigma^\frown\infty$-block and $\sigma^\frown f$-block to correspond to the β_σ-block and α_σ-block respectively. These blocks will contain classes denoted by $[t^n_{\gamma,j}]_T$ and $[b^n_{\gamma,j}]_B$ where γ is either $\sigma^\frown\infty$ or $\sigma^\frown f$ and $j \in \omega$.

Let $\tau_0, \tau_1, \ldots, \tau_k$ denote the terminal nodes in lexicographic order in Γ_n. Wait for $\varphi_{\tau_i}(t^n_0)\downarrow$ for each $i \leq k$. Now for each σ of even length and $|\sigma| < 2n$, we wait for $q^{-1}_\sigma(t^n_0)\downarrow$. Consider the following possibilities.

(1) Wait for $\varphi_{\tau_i}(t^n_0)\downarrow$ for each $\tau_i \in l^\beta_\sigma$. Once this happens, if $q^{-1}_\sigma(t^n_0)$ has converged, then enumerate $[t^n_{\sigma^\frown\infty,0}]_T$ (and $[b^n_{\sigma^\frown\infty,0}]_B$) into T (and B). Whenever $q^{-1}_\sigma(t^n_{\sigma^\frown\infty,j})\downarrow$, enumerate $[t^n_{\sigma^\frown\infty,j+1}]_T$ (and $[b^n_{\sigma^\frown\infty,j+1}]_B$) into T (and B).

(2) Wait for $\varphi_{\tau_i}(t^n_0)\downarrow$ for each $\tau_i \in l^\alpha_\sigma$. Once this happens, if $q^{-1}_\sigma(t_0)$ has not converged or $\varphi_{\tau_i}(t^n_0)\downarrow \neq b_{\sigma^\frown\infty,j}$ for any j and any $\tau_i \in l^\alpha_\sigma$, then enumerate the class $[t_{\sigma^\frown f,0}]_T$ (and $[b_{\sigma^\frown f,0}]_B$) into T (and B). Otherwise, do nothing; we do not build the $\sigma^\frown f$-block.

Once $\varphi_{\tau_k}(t^n_0)\downarrow = b^n_{\gamma,j}$ for some τ_k and $\gamma \prec \tau_k$, enumerate $[b^n_0]_B$ into B, close all blocks except the γ-block, assign Q_{σ_k} to this gadget and declare it prepared. Define $\mathbf{b}^n$ as the collection of elements containing b^n_0 and $b^n_{\xi,j}$ where $\xi \neq \gamma$. Since every block except the γ-block is closed, $|\mathbf{b}^n|$ is finite. Tag $[b^n_0]_B$ and all classes within the closed blocks with $G_1(N)$ for some N sufficiently large. For

the classes within the γ-block which is open, we tag each class with $G_1(y)$ for some appropriately chosen $y > N$.

In general, after a gadget is prepared, the sizes of the classes within the gadget should satisfy the following conditions.

- If γ is such that the γ-block is open, then for any $b \in B$ or $t \in T$ within a ξ-block where R_{ξ^-} is of higher priority than R_{γ^-}, $\#[b]^s_B = \#[t]^s_T = \#[t^n_0]^s_T$ for all stages s.
- Suppose that $\xi = \sigma^\frown\infty$ and the ξ-block is closed. If in addition $\alpha_\sigma(t^n_0) = q^{-1}_\sigma(t^n_0)$, then for each $b \in B$ in the ξ-block, $\#[b]^s_B = \#[h(b)]^s_T = \#[q^{-1}_\sigma(t^n_0)]^s_{A_\sigma}$. Otherwise, let $\#[b]^s_B = \#[h(b)]^s_T = \#[t^n_0]^s_T$ for each $b \in B$ in the ξ-block.
- If $\xi = \sigma^\frown f$ and the ξ-block is closed, then for each $b \in B$ within the ξ-block, let $\#[b]^s_B = \#[h(b)]^s_T = \#[t^n_0]^s_T$.
- If γ is such that the γ-block is open, then for each class within the γ-block, it follows the size of its tag.

4.3.2. *The diagonalisation procedures.* Throughout the discussion thus far, we have stated a few different properties that we require the various $q_\sigma p_\sigma$ and φ_γ to satisfy during the construction. We restate them here.

(D1) For any $t \in T$ in the n^{th} gadget and γ a terminal node of Γ_n, $\varphi_\gamma(t)$ must also be contained within the n^{th} gadget.

(D2) If Q_γ was assigned to the n^{th} gadget, then as long as some block remains open within the n^{th} gadget, $\mathbf{b}^n \not\subseteq \mathrm{rng}(\varphi_\gamma)$.

(D3) For any $b \in B$ in the n^{th} gadget and $\sigma \in \Gamma_n$ where $|\sigma| = 2e$ for some e, $q_\sigma p_\sigma(b)$ must also be contained within the n^{th} gadget.

(D4) If the n^{th} gadget is prepared with the γ-block left open, then for any $\sigma \in \Gamma_n$ of even length where $\sigma \prec \gamma^-$ or $\sigma \subsetneq \gamma^-$, $(q_\sigma p_\sigma)^{-1}(t^n_0)\downarrow = b$ where $b =_B b^n_0$ or b is contained within a ξ-block for some $\xi \prec \sigma$ or $\xi \subsetneq \sigma$.

(D5) If the n^{th} gadget has not been prepared, then for any $\sigma \in \Gamma_n$ of even length, $(q_\sigma p_\sigma)^{-1}(t^n_0)\uparrow$. In addition, once the gadget becomes prepared, if $(q_\sigma p_\sigma)^{-1}(t^n_0)\downarrow =_B b$, then b must be contained in some closed block of the n^{th} gadget.

We shall discuss here how we diagonalise against $q_\sigma p_\sigma$ or φ_γ should any of the conditions above fail to be met. The specific diagonalisation strategy shall differ slightly depending on G_1, G_2 and so we split the procedures into two cases.

For all y, $G_1(y) = M_1$ and $G_2(y) = M_2$: Non-uniformly fix M_1, M_2 and consider the following scenarios.

- Suppose that there is some $t \in T$ within the n^{th} gadget and some terminal $\gamma \in \Gamma_n$ such that $\varphi_\gamma(t)\downarrow$ is not within the n^{th} gadget. If $[\varphi_\gamma(t)]_B$ is currently tagged with M_1, then we close all blocks within the n^{th} gadget and change all their tags to M_2. After some finite wait, φ_γ must be observed to fail.

If $[\varphi_\gamma(t)]_B$ is instead tagged with M_2, then we similarly close all blocks within the n^{th} gadget but leave all their tags at M_1. As long as the tags of the n^{th} gadget remains at M_1, φ_γ must fail to be surjective. To ensure this, we will only change the tags of the n^{th} gadget for the sake of some requirement in Γ_n.

- Let Q_σ be the requirement assigned to the n^{th} gadget and suppose that $\mathbf{b}^n \subseteq \mathrm{rng}(\varphi_\gamma)$. Recall that since $\varphi_\gamma(t_0^n) \neq_B b$ for any $b \in \mathbf{b}^n$, then by applying pigeonhole principle, there must be some $b \in \mathbf{b}^n$ for which $\varphi_\gamma^{-1}(b) \neq_T h(b')$ for any $b' \in \mathbf{b}^n$. We may further assume that $\varphi_\gamma^{-1}(b)$ is still within the n^{th} gadget, otherwise we simply close all blocks in the n^{th} gadget and apply pigeonhole principle again to conclude that φ_γ fails (D1). Then given that $\varphi_\gamma^{-1}(b)$ is within the n^{th} gadget and not in any closed block, we will then increase the tags of all classes within the sole open block of the n^{th} gadget to M_2 and wait for φ_γ to fail. This must happen in finite time since $\#[b]_B = M_1 < M_2 = \#[\varphi_\gamma^{-1}(b)]_T$. Furthermore, as the block that $\varphi_\gamma^{-1}(b)$ is contained in is still open, we are able to keep $\#[b]_B = M_1$ until after φ_γ is witnessed to fail. Once this is observed, then we close the n^{th} gadget and tag every class within it with M_2.

- If (D3) fails, then we apply the same procedure to diagonalise against $q_\sigma p_\sigma$ as in (D1).

- Suppose that (D4) fails; there exists some $\sigma \in \Gamma_n$ of higher priority than R_{γ^-} where the γ-block is left open, such that $(q_\sigma p_\sigma)^{-1}(t_0^n) \neq_B b$ for any b contained in ξ-blocks of higher priority than R_σ. Since the n^{th} gadget is assumed to have already been prepared with only the γ-block left open, it must be that all ξ-blocks where $\xi \prec \sigma$ or $\xi \subsetneq \sigma$ are closed. Let m be the number of distinct classes in B which are contained in some ξ-block within the n^{th} gadget. For each of these classes, we pick representatives, say $b_1, b_2, \ldots, b_m$ and compute $q_\sigma p_\sigma(b_0^n)$ and $q_\sigma p_\sigma(b_i)$ for each $1 \leq i \leq m$. By our assumption, none of these can be contained within $[t_0^n]_T$. That is to say, there is at least one $b \in \{b_0^n, b_1, b_2, \ldots, b_m\}$ such that $q_\sigma p_\sigma(b) =_T t$ where t is contained in some τ-block where $\tau \succ \sigma$ or $\tau \supsetneq \sigma$ (we may assume $q_\sigma p_\sigma$ satisfies (D3)). To diagonalise against $q_\sigma p_\sigma$, we will then tag $[t_0^n]_T$ and all classes within the ξ-blocks where R_{ξ^-} is of higher priority than R_σ with M_2 and keep the tags of all other classes within the n^{th} gadget at M_1. Such an action could threaten the primitive recursiveness of β_{τ^-} or α_{τ^-} since we are preventing the classes within τ-blocks of lower priority from growing while increasing $\#[t_0^n]_T$, but in return we obtain a satisfaction for R_σ where $\sigma \prec \tau^-$ or $\sigma \subseteq \tau^-$. Once R_σ is met via witnessing the failure of $q_\sigma p_\sigma$, we close the γ-block in the n^{th} gadget and tag all classes with M_2.

- If (D5) fails while the gadget is still not prepared, then all blocks within the n^{th} gadget must still be open. We may thus increase the size of every class except $[t_0^n]_T$ to M_2 and wait for $q_\sigma p_\sigma$ to fail. This must happen in finite time, and once it is witnessed, we enumerate $[b_0^n]_B$ into the gadget and close all blocks, tagging $[t_0^n]_T$ with M_2. The argument for when the gadget becomes prepared is similar, as we also obtain that $(q_\sigma p_\sigma)^{-1}(t_0^n)\downarrow$ must be contained in the open block.

A careful analysis of the procedures above allows us to conclude that once all blocks within a gadget is closed, the tags of the classes never again changes. Furthermore, the various actions are consistent with the conditions that the sizes of various classes should satisfy as mentioned in Section 4.3.1.

Not the previous case: Recall that G_1, G_2 are non-decreasing; we may thus assume in this case that G_2 is strictly increasing or $G_2(y) = \infty$ for all y. In particular, we may assume that $G_2(y) > y$ for all y.

- Once again, suppose that (D1) fails and let t and γ be such that $\varphi_\gamma(t)\downarrow$ is not within the n^{th} gadget. If $\varphi_\gamma(t)$ is in another gadget, say the m^{th}, temporarily pause the size of $[t_0^m]_T$, wait for all classes with representatives in $\mathbf{b}^m$ to reach the current size of $[t_0^m]_T$, then pause the classes in the open block within the m^{th} gadget if it exists. This is to ensure that this action does not damage the definition of any other maps being maintained on the m^{th} gadget. Once the sizes of classes within the m^{th} gadget have temporarily stabilised, say at x, we close the n^{th} gadget and tag all classes within the n^{th} gadget with $G_2(y)$ for some sufficiently large $y \geq x$. After a finite wait, φ_γ must be witnessed to fail as $\#[t]_T = G_2(y) > y \geq x = \#[\varphi_\gamma(t)]_B$. Once such a failure is witnessed, resume growing the classes within the m^{th} gadget.

- Suppose that Q_γ was assigned to the n^{th} gadget and that φ_γ satisfies (D1). If there is still some block which is open in the n^{th} gadget and $\mathbf{b}^n \subseteq \mathrm{rng}(\varphi_\gamma$, then applying pigeonhole principle allows us to conclude that there is some $b \in \mathbf{b}^n$ such that $\varphi_\gamma^{-1}(b) = t$ for some t within the open block. Temporarily pause the size of $[t_0^n]_T$, and tag all classes within the open block with $G_2(x)$ where x is the current size of $[t_0^n]_T$. After some finite wait, φ_γ must be witnessed to fail and we will then close the final open block in the n^{th} gadget and tag all classes within the n^{th} gadget with $G_2(y)$ for some sufficiently large $y \geq x$.

- If (D3) fails, we can diagonalise against $q_\sigma p_\sigma$ by applying the same procedure as if (D1) fails.

- If (D4) fails, then applying the same pigeonhole argument as before allows us to conclude that there exists b such that the following holds. $q_\sigma p_\sigma(b)$ is contained within some τ-block where $\tau^- \supseteq \sigma$ and either $b = b_0^n$ or b is contained within some closed ξ-block, where R_{ξ^-} is of higher priority than R_σ. By tagging $[t_0^n]_T$ and the classes within the ξ-blocks where R_{ξ^-} is of higher priority than R_σ with $G_2(y)$ for some large y, we must witness the failure of $q_\sigma p_\sigma$ after some finite delay provided we temporarily pause all other classes within the n^{th} gadget. Clearly, this action prevents certain τ-blocks from growing and would thus injure the strategies for R_{τ^-}, but all such τ is necessarily either such that $\tau^- \supseteq \sigma$ or $\tau^- \succ \sigma$.

- If (D5) fails, then we tag all classes within the n^{th} gadget except $[t_0^n]_T$ with $G_2(y)$ for some fresh y and wait for $q_\sigma p_\sigma$ to fail. Once it does, enumerate $[b_0^n]_B$, close all blocks within the n^{th} gadget and tag all classes within the n^{th} gadget with $G_2(y')$ for some new fresh y'.

4.4. **Formal construction.** We split the construction into stages; at every stage, a new gadget will be started and the stage only ends when the gadget becomes prepared. That is, a single stage need not last only for a primitive recursive amount of time. In order to keep B, T primitive recursive, recall that we have some fixed primitive recursive presentation of E. Whenever the fixed presentation enumerates some element or class, we enumerate a new class of size one into the portion of the structure dedicated to padding. This ensures that B, T are kept primitive recursive.

At each stage n of the construction, we will maintain a finite subtree Γ_s of Γ and a finite path δ_s contained in Γ_s. Recursively define Γ_s as follows. $\Gamma_0 = \{\langle\rangle, \langle\infty\rangle, \langle f\rangle\}$. Γ_s is the smallest set satisfying the following.

- Contains all nodes $\sigma \in \Gamma_{s-1}$ of higher priority than δ_s.
- Contains $\delta_s^\frown\infty$ if $|\delta_s|$ is odd.
- Closed under the property "if $\sigma \in \Gamma_s$ and $|\sigma|$ is even, then $\sigma^\frown\infty$ and $\sigma^\frown f$ are also in Γ_s".

At each stage s of the construction, we will define δ_s and proceed to stage $s+1$ once δ_s is defined. In addition, once δ_s has been defined, close all blocks in each gadget assigned with some lower priority Q_γ.

Nodes which are currently contained in Γ_s are termed as *active*, and all other nodes are *inactive*. The active nodes σ may additionally be in one of the following states.

- Satisfied.
- Assigned to some gadget, provided σ is of odd length. For convenience, we will use γ_m to denote the node for which Q_{γ_m} is assigned to the m^{th} gadget.

During the construction, we say that the n^{th} gadget requires attention if one of (D1), (D2), (D3), or (D4) is witnessed to fail within the n^{th} gadget via some currently active and unsatisfied node, and the gadget still has some open block.

Stage $s \geq 0$**:** Let Γ_s be given and suppose that we have defined $\delta_{s'}$ for each $s' < s$ and the m^{th} gadget for each $m < n$. The stage will be split into three main phases. In the first phase, we increase the sizes of each class currently enumerated where necessary. During the second phase, we attend to each gadget requiring attention. Finally, we prepare the n^{th} gadget in the third phase.

Phase 1: Let $m < n$ be given. If all blocks in the m^{th} gadget is already closed, then let the size of each class within the m^{th} gadget grow to the current size of their tags. Similarly, each class within a block associated with a currently inactive strategy will also simply copy its tag.

If there is some block in the m^{th} gadget still open, then let $\#[t_0^m]_T^n$ and $\#[b_0^m]_B^n$ be the current size of their tags. For the remaining classes within the m^{th} gadget, consider the following cases.

- If $[b]_B, [h(b)]_T$ are such that b (and $h(b)$) is contained in a closed σ-block where σ of lower priority than the open block of the m^{th} gadget, and $\sigma = \sigma^- {}^\frown\infty$, then let $\#[b]_B^n$ and $\#[h(b)]_T^n$ be the size of $\#[q_{\sigma^-}^{-1}(t_0^m)]_{A_{\sigma^-}}^n$.
- Otherwise, let $\#[b]_B^n, \#[h(b)]_T^n$ be the current size of their tags.

For the m^{th} gadget, wait until $\varphi_\gamma(t)\downarrow$ for each newly enumerated $t \in [t_0^m]_T$ and for each terminal $\gamma \in \Gamma_{s_m}$ (where s_m is the stage the m^{th} gadget was prepared) and $\gamma \prec \gamma_m$ or $\gamma = \gamma_m$. Once this finite wait is over, increase the sizes of all remaining classes which do not yet match their tags. If in addition, a diagonalisation against some φ_γ was obtained, then declare γ satisfied and let $\delta_s = \gamma$.

Phase 2: If there are no gadgets which require attention, proceed to the next phase. Otherwise, suppose that the m^{th} gadget requires attention; there is some active σ which is witnessed to fail one of (D1), (D2), (D3), or (D4) within the m^{th} gadget. If there are multiple such gadgets, then we act for the highest priority σ. Once we have applied the appropriate actions as described in Section 4.3.2, declare σ satisfied and let $\delta_s = \sigma$. Note that this causes all other requirements which possibly required attention to become inactive.

Phase 3: Let $\tau_0 \prec \tau_1 \prec \ldots \prec \tau_k$ be the current terminal nodes of Γ_s (after the various actions in Phases 1 and 2, Γ_s could be different than it was at the beginning of the stage). Enumerate the free class $[t_0^n]_T$ to begin preparing the n^{th} gadget. In the lexicographic order of τ_i, we compute $\varphi_{\tau_i}(t_0^n)$. While computing $\varphi_{\tau_i}(t_0^n)$, we also check that for each σ of higher priority than τ_i and $|\sigma|$ is even, $q_\sigma p_\sigma(b)$ is contained within the n^{th} gadget for any b in the n^{th} gadget. Otherwise, σ fails to satisfy (D3) which allows us to obtain a diagonalisation against $q_\sigma p_\sigma$. Once such a diagonalisation is obtained, declare σ satisfied, close all blocks in the n^{th} gadget, let $\delta_s = \sigma$ and proceed to the next stage. Similarly, if $\varphi_{\tau_i}(t_0^n)\downarrow$ and is not in the n^{th} gadget (τ_i fails (D1)), then we apply the appropriate diagonalisation procedure, declare τ_i satisfied, close all blocks in the n^{th} gadget, let $\delta_n = \tau_i$ and proceed to the next stage. We may thus assume that for each τ_i and for each σ of higher priority than τ_i where $|\sigma|$ is even, τ_i and σ satisfies (D1) and (D3) respectively.

Wait for $\varphi_{\tau_i}(t_0^n)\downarrow = b$ where b is contained in a ξ-block for some $\xi \prec \tau_i$. To see that such a τ_i exists, note that for any $\sigma \subseteq \tau_k$ where $|\sigma|$ is even, the $\sigma^\frown f$-block will never be enumerated. That is, once $\varphi_{\tau_k}(t_0^n)\downarrow$, if it is contained within the n^{th} gadget, it has to be in a ξ-block where $\xi \prec \tau_k$ (τ_k is the right-most terminal node in Γ_s). Once the first such τ_i is found, let $\delta_s = \tau_i$, assign τ_i to the n^{th} gadget and close every block in the n^{th} gadget except the one containing $\varphi_{\tau_i}(t_0^n)$.

4.5. **Verification.** In the lemma that follows, f_s is as defined in Section 4.1.

Lemma 4.1. *If for all y, $G_2(y) < \infty$, then $f = \lim_s f_s$ exists and is a total isomorphism.*

Proof. We prove by induction that for each $i \in \omega$, $f([t_i]_T)\downarrow$. That is, for each i, there exists some s such that for all $s' \geq s$, $f_{s'}([t_i]_T) = f_s([t_i]_T)\downarrow$. First consider the case when $i = 0$. By assumption, $G_2(y) < \infty$ for all y. In particular, since the index of $[t_0]_T$ is even, it will have a tag in $\mathrm{rng}(G_1) \cup \mathrm{rng}(G_2)$. In addition, since we choose the tags for the even indexed classes, as long as we only change them finitely often, the tag of $[t_0]_T$ must eventually stabilise. Since G_1, G_2 are nice, then there must be some class in E which has the size of the tag of $[t_0]_T$. Let $[e_j]_E$ be such a class with the lowest index. Then fix a stage s such that $\#[t_0]_T^s = \#[e_j]_E^s = \#[t_0]_T$ which must exist as $\#[t_0]_T < \infty$. By the definition of f_s, it follows that $f_s([t_0]_T) = [e_j]_E$. Since $[t_0]_T$ is always the first class on which $f_{s'}$ is defined, and the sizes of $[e_j]_E$ and $[t_0]_T$ never change after stage s, then we obtain that $f_{s'}([t_0]_T) = [e_j]_E$ for all $s' \geq s$.

Suppose inductively that for each $i' < i$, $f([t_{i'}]_T)\downarrow$. Then there exists some stage n such that for all stages $s \geq n$, $f_s([t_{i'}]_T) = f_n([t_{i'}]_T)\downarrow$.

Case 1: i is odd. If $f_n([t_i]_T)\downarrow$, then we must have that $f_s([t_i]_T) = f_n([t_i]_T)$ for all $s \geq n$ as $f_s([t_{i'}]_T)$ for each $i' < i$ are already fixed after stage n. We may thus assume that $f_n([t_i]_T)\uparrow$. It thus remains to show that there is some class in E that $[t_i]_T$ can map to. If there is no stage $n' < n$ for which $f_{n'}([t_i]_T)\downarrow$, then it must be that $\#[t_i]_T^n = 1$, because the sizes of the odd

indexed classes always follow the sizes of their images under the current approximation of f. Since there are infinitely many classes of size at least 1 in E, then there must be some stage $s \geq n$ such that some fresh class $[e_j]_E$ of size at least 1 is enumerated. Then at such a stage s, $f_s([t_i]_T)\downarrow = [e_j]_E$.

Now suppose that there is some $n' < n$ such that $f_{n'}([t_i]_T)\downarrow$. We may further suppose that n' is such that for each n'' where $n' < n'' \leq n$, $f_{n''}([t_i]_T)\uparrow$. Recall that since $f_{n'}([t_i]_T)\downarrow$, then $f_{n'+1}([t_i]_T)\uparrow$ only if at stage $n'+1$, some class $[t_{2m}]_T$ where $2m < i$ is mapped to $f_{n'}([t_i]_T)$. That is to say, the tag of $[t_{2m}]_T$ at stage $n'+1$ is the same as the size of the tag of $[t_i]_T$ at stage $n'+1$. Then there must be infinitely many classes of size at least the current tag of $[t_i]_T$, as G_1 and G_2 are nice and non-decreasing and the tag of $[t_{2m}]_T$ always lies in $\mathrm{rng}(G_1) \cup \mathrm{rng}(G_2)$. Furthermore, after stage n', f will not be defined on any of the newly enumerated odd indexed classes until some image is found for $[t_i]_T$. Thus, at some sufficiently large stage $s \gg n$, there must be some class $[e_j]_E$ such that $[e_j]_E \neq f_{s-1}([t_{i'}]_T)$ for any i' odd, $\#[e_j]^s_E \geq \#[t_i]^s_T = \#[t_i]^{n'}_T$ and $[e_j]_E \neq f_s([t_{i'}]_T)$ for any $i' < i$ and even. Then such a class $[e_j]_E$ must be chosen as the image of $[t_i]_T$. Furthermore, as all even indexed classes $< i$ are fixed, then for all $s' \geq s$, we have that $f_{s'}([t_i]_T) = f_s([t_i]_T) = [e_j]_E$.

Case 2: i is even. Recall that each even class t_i will be tagged with $G_1(m)$ or $G_2(m)$ for some $m > 2i$. Furthermore, we will also ensure that the tag of each given class is changed only finitely often. By assumption that $G_2(y) < \infty$ for all y, then the limit of the tag of t_i must be some finite value N. Consider some stage s large enough after n such that the tag of $[t_i]_T$ currently matches N, and at least m many classes of size exactly N have shown up in the primitive recursive presentation of E and are also currently of size N. Such a stage s must exist as N is in the range of G_1 or G_2; there should be at least m classes of size N in E. At such a stage, we claim that $f_s([t_i]_T)\downarrow$. When defining $f_s([t_i]_T)$, we must ensure that $[t_i]_T$ maps to some $[e_j]_E$ of the same size (as the tag of t_i) where $j \geq i/2$ and $[e_j]_E \neq f_s([t_{i'}]_T)$ for any $i' < i$. We need then to count the maximum number of classes of size N which $[t_i]_T$ is not allowed to map to. Consider the worst case as follows. Each of the classes $[e_{j'}]_E$ where $j' < i/2$ is of size N. Furthermore $f_s([t_{i'}]_T)$ also maps to some class $[e_{j'}]_E$ of size N for all $i' < i$. That is, there are potentially $i + i = 2i$ many classes of size N which $[t_i]_T$ is not allowed to map to. However, we know that there are at least $m > 2i$ classes of size N. Thus there must be some $[e_j]_E$ which can serve as the image for $f_s([t_i]_T)$. After such a stage s, since $s \geq n$, and $\#[e_j]^s_E = \#[e_j]_E = N$, then for all $s' \geq s$, $f_{s'}([t_i]_T) = f_s([t_i]_T)\downarrow$.

By induction, we have that $\lim_s f_s(t_i)\downarrow$ for all $i \in \omega$. It thus remains to check that $f = \lim_s f_s$ is an isomorphism. If i is even, then at the stage where $f_s([t_i]_T)$ becomes fixed, we know that the size of $f_s([t_i]_T)$ is the same as the tag of $[t_i]_T$ at that stage. Since $[t_i]_T$ will eventually match the size of its tag, then the size of $[t_i]_T$ is the same as $f_s([t_i]_T)$. If i is odd, recall that the size of $[t_i]_T$ is kept the same as $f_s([t_i]_T)$ stagewise if $f_s([t_i]_T)\downarrow$. That is, for stages $s' \geq s$ after $f_s([t_i]_T)$ becomes fixed, the size of $[t_i]_T$ is the same as the size of $f_s([t_i]_T)$.

It follows directly from the definition of f that it is injective. To see that it is also surjective, let $[e_j]_E$ be some class in E. Fix some stage s such that $\#[e_j]^s_E = \#[e_j]_E$ and $\#[t_{2n}]^s_T = \#[t_{2n}]^s_T$ for all $2n \leq j$. If one of $[t_{2n}]_T$ is mapped to $[e_j]_E$ by f_s, then we have that $[e_j]_E \in \mathrm{rng}(f)$ as $f_s([t_{2n}]_T)$ must have become fixed by the choice of s. Suppose then that none of the even indexed classes $2n \leq j$ are mapped to $[e_j]_E$. Since $f_s([t_{2n}]_T)$ are all fixed, the next odd indexed class which maps

to $[e_j]_E$ will have its image fixed as $[e_j]_E$. Recall that only even indexed classes are able to remove another class from the domain of f_s. Since all the even indexed classes with the ability to change $f_{s'}^{-1}([e_j]_E)$ for $s' \geq s$ already have fixed images, then once $f_{s'}^{-1}([e_j]_E)\downarrow$ for some $s' \geq s$, it never again changes. Furthermore, by the definition of $f_{s'}$, some odd indexed class must eventually map to $[e_j]_E$. Thus f is surjective. □

Recall from Section 4.1 that if $G_2(y) = \infty$ for some y, then we do not actually require f to witness the bijection between E and T. Thus, in any case, we will construct equivalence structures T and B of the correct isomorphism type.

Lemma 4.2. *For each $b \in B$, $\#[b]_B$ is equal to the limit of its tag.*

Proof. The only classes which are possibly restricted are those of the form $[b^n_{\sigma^\frown\infty,j}]_B$. If the n^{th} gadget was prepared with the ξ-block left open, then for σ of lower priority than ξ^-, $[b^n_{\sigma^\frown\infty,j}]_B$ needs to be kept at the same size as $[q_\sigma^{-1}(t_0^n)]_{A_\sigma}$ provided $\alpha_\sigma(t_0^n)$ was defined to be $q_\sigma^{-1}(t_0^n)$. The challenge procedure (recall from Section 4.2.4) now serves another purpose; if the size of $[q_\sigma^{-1}(t_0^n)]_{A_\sigma}$ does not grow, then we are able to diagonalise against some φ_τ where $\tau \prec \sigma$, making σ inactive, and thus grow $[b^n_{\sigma^\frown\infty,j}]_B$ to the desired size. □

Applying Lemmas 4.1 and 4.2, we obtain that $B \cong E$, and since h is trivially an isomorphism, we also get that $T \cong E$. It remains to verify that each requirement is met.

Lemma 4.3. $\limsup_s |\delta_s| = \infty$.

Proof. We proceed by induction on m and show that for each m, there exists s such that $|\delta_s| \geq m$. More specifically, we construct an infinite string σ approximated by $\sigma_m = \sigma \upharpoonright m$ such that for each m, there are infinitely many s such that $\delta_s \supseteq \sigma_m$. Furthermore, there are only finitely many s for which $\delta_s \prec \sigma_m$.

Let $\sigma_0 = \langle\rangle$. It is easy to see that the desired property holds for such a choice of σ. Now suppose inductively that we have defined σ_m satisfying the desired property. We claim now that there exists infinitely many s for which $\delta_s \supseteq \sigma_m^\frown\infty$ or $\delta_s \supseteq \sigma_m^\frown f$. Suppose for a contradiction that there are only finitely many stages s for which $\delta_s \supseteq \sigma_m^\frown\infty$ or $\delta_s \supseteq \sigma_m^\frown f$. By the inductive hypothesis, this means that there are infinitely many stages s for which $\delta_s = \sigma_m$. Let these stages be given by $s_0 < s_1 < s_2 < \ldots$. We may further assume that for all $s \geq s_0$, δ_s does not strictly extend σ_m.

Let s_i be given. We shall show that there exists some $s > s_i$ for which $\delta_s \prec \sigma_m$. Referring the reader back to the construction, each node δ_s is chosen from Γ_{s-1} and if δ_s is not a terminal node of Γ_{s-1}, then δ_s must have required attention at stage $s-1$ and subsequently marked as satisfied thereafter. If δ_s never shifts left of δ_{s_i} for any $s > s_i$, then any node $\tau \subseteq \sigma_m = \delta_{s_i}$ never becomes inactive after stage s_i and will thus remain marked as satisfied. Since each node marked as satisfied can no longer require attention, as long as δ_s never becomes lexicographically left of δ_{s_i}, there can only be finitely many stages s for which $\delta_s \subseteq \delta_{s_i}$. But this contradicts our assumption that $\delta_{s_j} = \sigma_m$ for infinitely many $s_j > s_i$. That is, for each s_i, there exists $s > s_i$ where $\delta_s \prec \sigma_m$, which contradicts the inductive hypothesis. That is, there must be infinitely many stages s for which $\delta_s \supseteq \sigma_m^\frown\infty$ or $\delta_s \supseteq \sigma_m^\frown f$.

If there are infinitely many s for which $\delta_s \supseteq \sigma^\frown_m\infty$, then let $\sigma_{m+1} = \sigma^\frown_m\infty$. Otherwise, let $\sigma_{m+1} = \sigma^\frown_m f$. Observe that this ensures σ_{m+1} satisfies the property that there are infinitely many s for which $\delta_s \supseteq \sigma_{m+1}$. Furthermore, by choice of σ_{m+1}, if there are infinitely many s for which $\delta_s \prec \sigma_{m+1}$, then it follows that there are infinitely many s for which $\delta_s \prec \sigma_m$. □

As $\delta_s \in \Gamma_s$, it follows that $|\bigcup_s \Gamma_s| = \infty$ and that there is a path through $\bigcup_s \Gamma_s$. Let δ denote the left-most infinite path in $\bigcup_s \Gamma_s$, and we now show that along δ, each requirement is satisfied. (In fact $\delta = \sigma$ defined in the proof above.)

Lemma 4.4. *Let $\sigma \in \bigcup_n \Gamma_n$ of even length be given. If $\sigma^\frown\infty \subseteq \delta$, then l^β_σ becomes fixed after some finite stage s. Similarly, if $\sigma^\frown f \subseteq \delta$, then l^α_σ becomes fixed after some finite stage s.*

Proof. Fix a stage s such that for all $s' > s$, no nodes $\gamma \prec \delta$ are added to $\bigcup_n \Gamma_n$. Such a stage s must exist, as the collection $\gamma \in \bigcup_n \Gamma_n$ such that $\gamma \prec \delta$ must be finite as δ is the left-most path. It follows that l^β_σ (or l^α_σ) does not change after stage s. □

Lemma 4.5. *Let $\xi \in \bigcup_n \Gamma_n$ of odd length be such that $\xi \subseteq \delta$ or $\xi \succ \delta$. For any gadget in which the ξ-block is left open when it is prepared, there exists some later stage s where the ξ-block becomes closed.*

Proof. Let ξ be such that the n^{th} gadget was prepared with the ξ-block left open. Recall that this only happens if $\varphi_\tau(t^n_0)\downarrow =_B b^n_{\xi,j}$ for some $\tau \in \Gamma_n$ terminal and $\tau \succ \xi$. Suppose for a contradiction that the ξ-block within the n^{th} gadget never becomes closed. This implies that after the n^{th} gadget becomes prepared, δ_s is never of higher priority than τ for all subsequent stages s. This directly contradicts the assumption that $\xi \succ \delta$ or $\xi \subseteq \delta$. □

Definition 4.6. *Let $\sigma \in \bigcup_n \Gamma_n$ of even length be given. We define β_σ on each gadget separately. For each $a \in A_\sigma$ where $q_\sigma(a)$ is in a gadget prepared before this version of β_σ began, define $\beta_\sigma(a) = h^{-1}q_\sigma(a)$. We now consider a for which $q_\sigma(a)$ is in some gadget, say the n^{th}, started after this version of β_σ. While the n^{th} gadget is being prepared, define β_σ as follows.*

- *If $q_\sigma(a) =_T t^n_0$, then define $\beta_\sigma(a) =_B b^n_0$ if it exists (at the time that $\beta_\sigma(a)$ has to be defined). Otherwise, define $\beta_\sigma(a) =_B b^n_{\sigma^\frown\infty,0}$.*
- *If $q_\sigma(a) =_T t^n_{\sigma^\frown\infty,j}$ for some j, then define $\beta_\sigma(a) =_B b^n_{\sigma^\frown\infty,j+1}$.*
- *Otherwise, define $\beta_\sigma(a) =_B h^{-1}q_\sigma(a)$.*

Once the $\sigma^\frown\infty$-block of the n^{th} gadget is closed, let $[t^n_{\sigma^\frown\infty,k}]_T$ be the final class of the $\sigma^\frown\infty$ block (if it exists). Then define $\beta_\sigma(a) =_B b^n_0$ for the a such that $q_\sigma(a) =_T t^n_{\sigma^\frown\infty,k}$.

While the definition above only defines β_σ on classes, as long as we can prove that for the final version of β_σ, $\#[a]^s_{A_\sigma}$ is always at most $\#[\beta_\sigma(a)]^s_B$, and that $\#[a]_{A_\sigma} = \#[\beta_\sigma(a)]_B$, then β_σ is a primitive recursive isomorphism from A_σ onto B.

Recall that the intuition behind the definition of β_σ on a single gadget is that if the structure A_σ shows $q^{-1}_\sigma(t^n_0)$ 'quickly', in order to maintain the primitive recursiveness of β_σ, we enumerate the β_σ block into the n^{th} gadget. However, doing so would result in the temporary loss of surjectivity of

β_σ. If instead $q_\sigma^{-1}(t_0^n)$ is 'slow' to show up, we then attempt to make β_σ surjective, at least within this gadget. We provide the formal details below.

Lemma 4.7. *Let σ of even length be such that $\sigma^\frown\infty \subseteq \delta$. If $q_\sigma : A_\sigma \to_{onto} T$ is a primitive recursive isomorphism, then $\beta_\sigma : A_\sigma \to_{onto} B$ is a primitive recursive isomorphism.*

Proof. Fix some σ of even length such that $\sigma^\frown\infty \subseteq \delta$. Applying Lemma 4.4 allows us to conclude that l_σ^β is fixed after some stage, say s^*. In particular, after s^*, the version of β_σ is the final one. Also suppose that $l_\sigma^\beta = \{\tau_0, \tau_1, \ldots, \tau_m\}$. We split the proof into three main parts; proof of primitive recursiveness, surjectivity and injectivity.

Proof of primitive recursiveness. Let $a \in A_\sigma$ be given. Once a is enumerated into A_σ, wait for $q_\sigma(a)\downarrow$ (which must happen in primitive recursive time). We run through the possible cases in the definition of β_σ. On all classes $[t]_T$ enumerated before stage s^*, it must be that $[h^{-1}(t)]_B$ has also been enumerated. Therefore, if $q_\sigma(a) =_T t$ for some $[t]_T$ enumerated before stage s^*, then we are able to define $\beta_\sigma(a) =_B h^{-1}q_\sigma(a)$. In the cases that follow, we consider only the classes $[t]_T$ enumerated after stage s^*.

- If $q_\sigma(a) =_T t_0^n$, we claim that $\beta_\sigma(a)\downarrow$ before (or within some primitive recursive delay of) $\varphi_{\tau_m}(t_0^n)$ converging. If the gadget becomes prepared before $\varphi_{\tau_m}(t_0^n)\downarrow$, then $\beta_\sigma(a) =_B b_0^n$ the moment the gadget becomes prepared. We may thus suppose that $\varphi_{\gamma_m}(t_0^n)\downarrow$ before b_0^n is enumerated into B. Referring the reader back to Section 4.3.1, if $\varphi_{\tau_m}(t_0^n)\downarrow$ after $q_\sigma^{-1}(t_0^n)\downarrow$, then we would have enumerated $[b^n_{\sigma^\frown\infty,0}]_B$ into the n^{th} gadget, serving as the β_σ image for $[a]_{A_\sigma}$.
- If $q_\sigma(a) =_T t^n_{\sigma^\frown\infty,j}$ for some j and the $\sigma^\frown\infty$-block is not yet closed, then $b_{\sigma^\frown\infty,j+1}$ must have been enumerated into B the moment $q_\sigma^{-1}(t^n_{\sigma^\frown\infty,j})\downarrow$. On the other hand, if $q_\sigma(a) =_T t^n_{\sigma^\frown\infty,j}$ where j is the largest index in the $\sigma^\frown\infty$-block and the block has closed, we define $\beta_\sigma(a) =_B b_0^n$. We know that $[b_0^n]_B$ has been enumerated by the time that the $\sigma^\frown\infty$-block closes as this only happens after the gadget is prepared.
- Finally, if none of the previous cases hold, then $q_\sigma(a) \neq_T t_0^n$ and $q_\sigma(a) \neq_T t^n_{\sigma^\frown\infty,j}$ for any j. That is, we have the property that $[h^{-1}q_\sigma(a)]_B$ is always enumerated at the same time as $[q_\sigma(a)]_T$, and $[h^{-1}q_\sigma(a)]_B$ is not yet in the range of β_σ. Thus we are able to let $\beta_\sigma(a) =_B h^{-1}q_\sigma(a)$.

We now have that β_σ is primitive recursive as a map from the equivalences classes of A_σ to the equivalence classes of B. It remains to prove that $\#[a]^s_{A_\sigma} \leq \#[\beta_\sigma(a)]^s_B$. Observe that except $t = t_0^n$ for some n where the n^{th} gadget is started after s^*, we have that for all $s \geq s^*$, $\#[t]^s_T = \#[h^{-1}(t)]^s_T$. In particular, if $\beta_\sigma(a) =_B h^{-1}q_\sigma(a)$ ($q_\sigma(a) \neq_T t_0^n$), then we know that $\#[a]^s_{A_\sigma} \leq \#[q_\sigma(a)]^s_T = \#[h^{-1}q_\sigma(a)]^s_T$ or q_σ cannot be primitive recursive and injective. On the other hand, if $\beta_\sigma(a) \neq_B h^{-1}q_\sigma(a)$, then it follows that $\beta_\sigma(a)$ is in the $\sigma^\frown\infty$-block of the n^{th} gadget. Recall that all classes within such a block are either kept at the same size as $\#[q_\sigma^{-1}(t_0^n)]^s_{A_\sigma}$ or kept at the same size as $\#[t_0^n]^s_T$. Thus, for any $a \in A_\sigma$ where $q_\sigma(a) =_T t_0^n$ or $q_\sigma(a) =_T t^n_{\sigma^\frown\infty,j}$ for some j, it follows that $\#[a]^s_{A_\sigma} \leq \#[q_\sigma(a)]^s_{A_\sigma} \leq \#[\beta_\sigma(a)]^s_B$.

Proof of surjectivity. We first prove that β_σ is surjective on classes, and then argue that for each $a \in A_\sigma$, $\#[\beta_\sigma(a)]_B = \#[a]_{A_\sigma}$. Let $[b]_B$ be given. If $[b]_B$ is not contained within any gadget, once $h(b)$ enters $\mathrm{rng}(q_\sigma)$, then we would have defined $\beta_\sigma(a) =_B b$ for the a such that $q_\sigma(a) =_T h(b)$.

Furthermore, in such a case, provided that q_σ is surjective, we will then obtain that $\#[a]_{A_\sigma} = \#[q_\sigma(a)]_T = \#[h(b)]_T = \#[b]_B$. That is, for any element b which is not contained in a gadget, $b \in \mathrm{rng}(\beta_\sigma)$. The argument for elements contained within some gadget started before stage s^* is similar.

We may thus suppose now that $[b]_B$ is contained within some gadget started at some stage $s \geq s^*$ (recall s^* such that $\delta_s \supseteq \sigma^\frown\infty$ or $\delta_s \succ \sigma^\frown\infty$ for all $s \geq s'$). In addition to this, we may further assume that b is of the form b_0^n, or $b^n_{\sigma^\frown\infty,j}$ for some n. If b is not of the form b_0^n or in the $\sigma^\frown\infty$-block, then applying a similar argument to before, once $q_\sigma(a)\downarrow =_T h(b)$ for some a, $\beta_\sigma(a)\downarrow =_B b$ and b eventually enters $\mathrm{rng}(\beta_\sigma)$. Recall from Definition 4.6 that once $q_\sigma(a)\downarrow =_T t^n_{\sigma^\frown\infty,j}$ for some j, then we would have defined $\beta_\sigma(a) =_B b^n_{\sigma^\frown\infty,j+1}$. Similarly, once $q_\sigma(a)\downarrow =_T t^n_0$, then we would have defined $\beta_\sigma(a) =_B b^n_{\sigma^\frown\infty,0}$. In summary, for each j, there exists some $a \in A_\sigma$ for which $\beta_\sigma(a) =_B b^n_{\sigma^\frown\infty,j}$, provided q_σ is surjective.

Recall the intuition that if the $\sigma^\frown\infty$-block is not closed, then β_σ will not be surjective. Applying Lemma 4.5 together with the assumption that $\sigma^\frown\infty \subseteq \delta$ allows us to conclude that the $\sigma^\frown\infty$-block must eventually become closed. Once the $\sigma^\frown\infty$-block becomes closed, then we would have defined $\beta_\sigma(a) =_B b^n_0$ for the $a \in A_\sigma$ where $q_\sigma(a)\downarrow =_T t^n_{\sigma^\frown\infty,k}$ and $[t^n_{\sigma^\frown\infty,k}]_T$ is the final class within the $\sigma^\frown\infty$-block. That is, after the $\sigma^\frown\infty$-block closes, provided q_σ is surjective, we must eventually define $\beta_\sigma(a) =_B b^n_0$. Furthermore, after the block is closed, all classes within it are kept at size either $\#[q_\sigma^{-1}(t^n_0)]^s_T$ or $\#[t^n_0]^s_T$. Provided that q_σ is surjective, this would imply that all classes within the $\sigma^\frown\infty$-block must have size $\#[t^n_0]_T$ in the limit.

Proof of injectivity. From before, we already have that for any $a \in A_\sigma$, $\#[a]^s_{A_\sigma} \leq \#[\beta_\sigma(a)]^s_B$. Provided that β_σ is injective on classes, then it follows that β_σ can also be made injective on elements. Suppose that $\beta_\sigma(a) =_B \beta_\sigma(a')$. We run through the following possibilities.

- Let n be such that the n^{th} gadget was started after stage s^*. If $\beta_\sigma(a) =_B \beta_\sigma(a') =_B b^n_{\sigma^\frown\infty,j}$ for some j, then either $q_\sigma(a) =_T q_\sigma(a') =_T t^n_0$ or $q_\sigma(a) =_T q_\sigma(a') =_T t^n_{\sigma^\frown\infty,j-1}$. If q_σ is injective, we thus obtain that $a =_{A_\sigma} a'$.
- If $\beta_\sigma(a) =_B \beta_\sigma(a') =_B b^n_0$ for some n where the n^{th} gadget is started after stage s^*, then we have $q_\sigma(a) =_T q_\sigma(a') =_T t^n_{\sigma^\frown\infty,k}$ where k is the largest index in the $\sigma^\frown\infty$-block within the n^{th} gadget, or $q_\sigma(a) =_T q_\sigma(a') =_T t^n_0$ (if the $\sigma^\frown\infty$-block does not exist in the n^{th} gadget). We may thus obtain that $a =_{A_\sigma} a'$ provided that q_σ is injective.
- If neither of the previous cases hold, then $q_\sigma(a) =_T q_\sigma(a') =_T h\beta_\sigma(a)$. Once again, this implies that $a =_{A_\sigma} a'$.

Since β_σ is surjective and injective on both the classes and elements, it follows that it is an isomorphism. □

Definition 4.8. *Let $\sigma \in \bigcup_n \Gamma_n$ of even length be given. We define α_σ on each gadget separately. For each $t \in T$ in a gadget prepared before this version of α_σ began, define $\alpha_\sigma(t) = p_\sigma h^{-1}(t)$. Now let $t \in T$ be in some gadget, say the n^{th}, which was started after this version of α_σ. While the n^{th} gadget is being prepared, define α_σ as follows.*

- *If $t = t_0^n$, then wait for $\varphi_\gamma(t_0^n)\downarrow$ for all $\tau \in l_\sigma^\alpha$. If at least one of the τ is such that $\varphi_\tau(t_0^n)\downarrow =_B b^n_{\sigma^\frown\infty,j}$ for some j, then define $\alpha_\sigma(t_0^n) =_{A_\sigma} q_\sigma^{-1}(t_0^n)$. Otherwise, define $\alpha_\sigma(t_0^n) =_{A_\sigma} p_\sigma(b^n_{\sigma^\frown f,0})$.*
- *If $t = t^n_{\sigma^\frown f,j}$ for some j, then define $\alpha_\sigma(t) =_{A_\sigma} p_\sigma(b^n_{\sigma^\frown f,j+1})$.*
- *Otherwise, define $\alpha_\sigma(t) =_{A_\sigma} p_\sigma h^{-1}(t)$.*

Once the n^{th} gadget becomes prepared, if the $\sigma^\frown f$-block exists within the n^{th} gadget and becomes closed, let $[t^n_{\sigma^\frown f,k}]_T$ be the final class within the block, then define $\alpha_\sigma(t^n_{\sigma^\frown f,k}) =_{A_\sigma} p_\sigma(b_0^n)$. On the other hand, if the $\sigma^\frown f$-block was not enumerated ($\alpha_\sigma(t_0^n) =_{A_\sigma} q_\sigma^{-1}(t_0^n)\downarrow$), and t is such that $t \neq_T t_0^n$ but $p_\sigma h^{-1}(t) =_{A_\sigma} q_\sigma^{-1}(t_0^n)$, then define $\alpha_\sigma(t) =_{A_\sigma} p_\sigma(b_0^n)$.

Lemma 4.9. *Let σ be of even length. If $\sigma^\frown f \subseteq \delta$, $p_\sigma : B \to_{onto} A_\sigma$ and $q_\sigma : A_\sigma \to_{onto} T$ are both primitive recursive isomorphisms, then $\alpha_\sigma : T \to_{onto} A_\sigma$ is a primitive recursive isomorphism.*

Proof. Fix some σ of even length such that $\sigma^\frown f \subseteq \delta$. Applying Lemma 4.4 allows us to conclude that l_σ^α is fixed after some stage s^*, and α_σ never gets initialised again after s^*. Let $l_\sigma^\alpha = \{\tau_0, \tau_1, \ldots, \tau_m\}$ be the stable state of l_σ^α. We split the proof into three main parts as before; proof of primitive recursiveness, surjectivity and injectivity.

Proof of primitive recursiveness. Let $t \in T$ be given. It is easy to see that if $\alpha_\sigma(t)$ was defined to be $p_\sigma h^{-1}(t)$, then $t \neq_T t_0^n$ for any n and the classes $[t]_T$ and $[h^{-1}(t)]_B$ would have been enumerated into B and T respectively at the same time. Thus, for such classes, $\#[t]_T^s = \#[h^{-1}(t)]_B^s \leq \#[p_\sigma h^{-1}(t)]_{A_\sigma}^s$; α_σ is primitive recursive on all elements contained in such classes. In the cases that follow, we consider the classes on which α_σ is not defined to be $p_\sigma h^{-1}$. In particular, t is of the form t_0^n or $t^n_{\sigma^\frown f,j}$ for some j, or t is such that $\alpha_\sigma(t) =_{A_\sigma} p_\sigma(b_0^n)$.

Recall that when defining $\alpha_\sigma(t_0^n)$, we are allowed to wait until $\varphi_{\tau_i}(t_0^n)\downarrow$ for all $\tau_i \in l_\sigma^\alpha$. We then have the following possibilities.

- There is some i such that $\varphi_{\tau_i}(t_0^n)\downarrow =_B b^n_{\sigma^\frown\infty,j}$ for some n. In order for this to hold, it must be that $q_\sigma^{-1}(t_0^n)\downarrow$, otherwise we would not have enumerated any class of the form $[b^n_{\sigma^\frown\infty,j}]_B$ into the n^{th} gadget. This implies that we are able to successfully define $\alpha_\sigma(t_0^n) =_{A_\sigma} q_\sigma^{-1}(t_0^n)$. It remains to check that at each stage $s \geq s^*$, $\#[t_0^n]_T^s \leq \#[\alpha_\sigma(t_0^n)]_{A_\sigma}^s$. In particular, we need only check that it holds after the n^{th} gadget becomes prepared (recall that all classes are of size 1 while the gadget is being prepared).

  Suppose that the gadget becomes prepared with the ξ-block left open. If σ is of higher priority than ξ^-, or if $\sigma = \xi^-$, then provided (D4) holds, we obtain that $(q_\sigma p_\sigma)^{-1}(t_0^n)\downarrow =_B b$ for some b where $\#[b]_B^s = \#[t_0^n]_T^s$ for all $s \geq s^*$. On the other hand, if σ is of lower priority than ξ^-, then all classes of the form $[b^n_{\sigma^\frown\infty,j}]_B$ are kept at the same size as $[q_\sigma^{-1}(t_0^n)]_{A_\sigma}$ stagewise. That is to say, if $\#[t_0^n]_T^s > \#[q_\sigma^{-1}(t_0^n)]_{A_\sigma}^s = \#[b^n_{\sigma^\frown\infty,j}]_B^s$, then we obtain that $\#[t_0^n]_T^s > \#[\varphi_{\tau_i}(t_0^n)]_B^s$ for the $\varphi_{\tau_i} \in l_\sigma^\alpha$ which converged to some element in the $\sigma^\frown\infty$-block. By the time the delay allowed for α_σ is up, either $[q_\sigma^{-1}(t_0^n)]_{A_\sigma}$ has grown, or we obtain a diagonalisation against some $\tau_i \prec \sigma$. But the latter cannot happen since we assumed that δ_s never goes left of $\sigma^\frown f$ for $s \geq s^*$.

If the gadget becomes prepared with all blocks closed, then all classes within this gadget will be kept at the same size. That is to say, as long as (D3) holds, we obtain that there exists b within the n^{th} gadget such that $q_\sigma^{-1}(t_0^n) =_{A_\sigma} p_\sigma(b)$ and $\#[b]_B^s = \#[t_0^n]_T^s$.

- Not the previous case: none of $\varphi_{\tau_i}(t_0^n)$ converged to some element within the $\sigma^\frown\infty$-block. Recall from Section 4.3.1 that if this happens, then we would enumerate the class $[b^n_{\sigma^\frown f,0}]_B$ into B. Furthermore, the p_σ image of such a class will serve as the α_σ image for t_0^n. Since the classes within any γ-block where $\gamma = \gamma^- {}^\frown f$ is kept to be the same size as $[t_0^n]_T$, we thus obtain that $\#[t_0^n]_T^s = \#[b^n_{\sigma^\frown f,0}]_B^s \leq \#[p_\sigma(b^n_{\sigma^\frown f,0})]_{A_\sigma}^s$ for all $s \geq s^*$.

If $t =_T t^n_{\sigma^\frown f,j}$ for some j, then provided that the $\sigma^\frown f$-block within the n^{th} gadget is not yet closed, we would have enumerated $[b^n_{\sigma^\frown f,j+1}]_B$ into B, thus providing the α_σ image for t via $p_\sigma(b^n_{\sigma^\frown f,j+1})$. Once the $\sigma^\frown f$-block closes, we would define $\alpha_\sigma(t^n_{\sigma^\frown f,k}) =_{A_\sigma} p_\sigma(b_0^n)$. Using the same argument as before, since all classes within the $\sigma^\frown f$-block is always kept at the same size as $[t_0^n]_T$, then we have that on all such elements, $\#[t]_T^s \leq \#[\alpha_\sigma(t)]_{A_\sigma}^s$.

Finally we consider the case that $\alpha_\sigma(t_0^n) =_{A_\sigma} q_\sigma^{-1}(t_0^n)$ and t is such that $t \neq_T t_0^n$ but $p_\sigma h^{-1}(t) =_{A_\sigma} \alpha_\sigma(t_0^n)$. For such a class $[t]_T$, we would like to define $\alpha_\sigma(t) =_{A_\sigma} p_\sigma(b_0^n)$. It thus remains to argue that should such a class $[t]_T$ be found, the n^{th} gadget must have become prepared. This follows directly from ensuring that (D5) holds during the construction. If (D5) is discovered to fail for σ at stage s, then we would have declared $\delta_s = \sigma$. Applying the assumption that for all stages $s \geq s^*$, $\delta_s \supseteq \sigma^\frown f$ or $\delta_s \succ \sigma^\frown f$, we obtain that (D5) will never be discovered to fail for σ after stage s^*. Furthermore, such a t must be contained within some closed block of the gadget, allowing us to obtain $\#[t]_T^s \leq \#[t_0^n]_T^s = \#[b_0^n]_B^s \leq \#[p_\sigma(b_0^n)]_{A_\sigma}^s$.

Proof of surjectivity. We first show that α_σ is surjective on the classes, and then show that for each t, $\#[t]_T = \#[\alpha_\sigma(t)]_{A_\sigma}$. Provided that p_σ is surjective, there exists $b \in B$ such that $p_\sigma(b) =_{A_\sigma} a$. We consider the following possibilities for b.

If b is not contained within any gadget, then we know that $\alpha_\sigma h(b) =_{A_\sigma} p_\sigma(b)$. Furthermore, we have that $\#[h(b)]_T = \#[b]_B$, and if p_σ is surjective and an isomorphism, it must also be that $\#[b]_B = \#[p_\sigma(b)]_{A_\sigma} = \#[a]_{A_\sigma}$. Thus, for all $a \in A_\sigma$ where $p_\sigma^{-1}(a)$ is not contained within any gadget, $a \in \mathrm{rng}(\alpha_\sigma)$.

We may thus suppose that a is such that $b \coloneqq p_\sigma^{-1}(a)$ is contained within some gadget. We may further assume that this gadget was started after stage s^*, otherwise b will have the property that $\alpha_\sigma h(b) =_{A_\sigma} p_\sigma(b)$ and the previous argument applies. Let this gadget be the n^{th} and suppose that the $\sigma^\frown f$-block was not enumerated into the n^{th} gadget ($\alpha_\sigma(t_0^n) =_{A_\sigma} q_\sigma^{-1}(t_0^n)$). We consider the different possible b within the n^{th} gadget.

- If b is such that $p_\sigma(b) =_{A_\sigma} q_\sigma^{-1}(t_0^n)$, then we have that $\alpha_\sigma(t_0^n) =_{A_\sigma} a$. Using the assumption that q_σ is an isomorphism, we have that $\#[t_0^n]_T = \#[q_\sigma^{-1}(t_0^n)]_{A_\sigma}$; $[a]_{A_\sigma} \subseteq \mathrm{rng}(\alpha_\sigma)$.

- If $b =_B b_0^n$, then provided that p_σ is surjective and (D5) holds, there must be some b' within the n^{th} gadget for which $p_\sigma(b') =_{A_\sigma} q_\sigma^{-1}(t_0^n)$. Once such a b' is found, we would have defined $\alpha_\sigma h(b') =_{A_\sigma} p_\sigma(b_0^n)$. Provided that (D5) holds, such a b' must come from one of the closed blocks within the n^{th} gadget. Applying Lemma 4.2, we obtain that $\#[h(b')]_T = \#[b_0^n]_B$ which in turn must be equal $\#[p_\sigma(b_0^n)]_{A_\sigma}$ provided p_σ is an isomorphism.

- If neither of the previous cases hold, then it follows that $\alpha_\sigma h(b) =_{A_\sigma} p_\sigma(b)$. It is easy to see that $\#[h(b)]_T = \#[b]_B = \#[p_\sigma(b)]_{A_\sigma}$, where the last equality follows from the assumption that p_σ is an isomorphism.

Now consider the possibility that the $\sigma^\frown f$-block was enumerated into the n^{th} gadget. This means that within the n^{th} gadget, we did not define $\alpha_\sigma(t_0^n) =_{A_\sigma} q_\sigma^{-1}(t_0^n)$. Since $\sigma^\frown f \subseteq \delta$, then by Lemma 4.5, there must be some finite stage at which the $\sigma^\frown f$-block becomes closed. Once it does, let $[b^n_{\sigma^\frown f,k}]_B$ be the final class in the $\sigma^\frown f$-block. Following Definition 4.8, we know that

- for each $j \geq 1$, $\alpha_\sigma(t^n_{\sigma^\frown f,j-1}) =_{A_\sigma} p_\sigma(b^n_{\sigma^\frown f,j})$,
- $\alpha_\sigma(t^n_{\sigma^\frown f,k}) =_{A_\sigma} p_\sigma(b^n_0)$,and
- $\alpha_\sigma(t^n_0) =_{A_\sigma} p_\sigma(b^n_{\sigma^\frown f,0})$.

That is, for the $a \in A_\sigma$ such that $p_\sigma^{-1}(a) =_B b$ where $b = b_0^n$ or is contained within the $\sigma^\frown f$-block, there exists $t \in T$ such that $\alpha_\sigma(t) =_{A_\sigma} a$. Since all classes within the $\sigma^\frown f$-block will be kept at the same size stagewise as $\#[t_0^n]^s_T$, we also obtain that for all such $a \in A_\sigma$, $[a]_{A_\sigma} \subseteq \mathrm{rng}(\alpha_\sigma)$. Thus, regardless of whether $\sigma^\frown f$-block was enumerated into the n^{th} gadget or not, for each $a \in A_\sigma$ where $p_\sigma^{-1}(a)$ is in the n^{th} gadget, $a \in \mathrm{rng}(\alpha_\sigma)$.

Proof of injectivity. Since we already have that for each $t \in T$ and for each stage $s \geq s^*$, $\#[t]^s_T \leq \#[\alpha_\sigma(t)]^s_{A_\sigma}$, it suffices to show that α_σ is injective on the classes. Suppose that t, t' are such that $\alpha_\sigma(t) =_{A_\sigma} \alpha_\sigma(t')$. If t, t' are not contained within any gadget or contained within some gadget prepared before s^*, then it follows that $\alpha_\sigma(t) =_{A_\sigma} p_\sigma h^{-1}(t)$ and $\alpha_\sigma(t') =_{A_\sigma} p_\sigma h^{-1}(t')$. Since p_σ is assumed to be an isomorphism, it follows that $t =_T t'$.

We may thus assume that t, t' are contained within some gadget started after stage s^*. It is not hard to see that if $t \neq_T t_0^n$ and $t' \neq_T t_0^n$, then $\alpha_\sigma(t) \neq_{A_\sigma} \alpha_\sigma(t')$, otherwise p_σ cannot be an isomorphism. It thus remains to check the possibility that $t =_T t_0^n$ but $t' \neq_T t$. In particular, it could potentially be an issue only if $\alpha_\sigma(t_0^n)$ was defined to be $q_\sigma^{-1}(t_0^n)$ and t' is such that $p_\sigma(t') =_{A_\sigma} q_\sigma^{-1}(t_0^n)$. However, recall that should this be discovered, we would have defined $\alpha_\sigma(t') =_{A_\sigma} p_\sigma(b_0^n)$. Furthermore, as $h(b_0^n) = t_0^n$, for no other t'' would $\alpha_\sigma(t'') =_{A_\sigma} p_\sigma(b_0^n)$.

Since α_σ is bijective on classes and for each $t \in T$, $\#[t]_T = \#[\alpha_\sigma(t)]_{A_\sigma}$, then α_σ is an isomorphism. □

Lemma 4.10. *If $\tau \subseteq \delta$ is of odd length, then Q_τ is satisfied.*

Proof. Since $\tau \subseteq \delta$, then there exists some stage s^* such that for all $s \geq s^*$, $\delta_s \supseteq \tau$ or $\delta_s \succ \tau$. In other words, past stage s^*, τ is always an active node (contained in Γ_s). Since $\delta \supseteq \tau$, then we know that the subtree extending τ in $\bigcup_n \Gamma_n$ must be infinite. In particular, there must be some stage $s \geq s^*$ such that $\delta_s = \tau$. By a simple analysis of the formal construction, we may then conclude that τ must have been placed in either the assigned state or the satisfied state during such a stage.

If Q_τ was assigned to the n^{th} gadget, then on the n^{th} gadget, $\mathbf{b}^n \nsubseteq \mathrm{rng}(\varphi_\tau)$. Otherwise, the n^{th} gadget would have required attention should the aforementioned property be found to fail. This then guarantees that we obtain a permanent diagonalisation against φ_τ, thus satisfying Q_τ. Either φ_τ is never surjective, or we obtain a diagonalisation against it at some finite stage. On the other

hand, if τ is ever placed in a satisfied state, a simple analysis of the formal construction allows us to conclude that a diagonalisation against it was obtained. □

References

[1] Nikolay Bazhenov, Iskander Kalimullin, Alexander Melnikov, and Keng Meng Ng. Online presentations of finitely generated structures. *Theoretical Computer Science*, 844:195–216, 2020.

[2] Wesley Calvert, Douglas Cenzer, Valentina Harizanov, and Andrei Morozov. Effective categoricity of equivalence structures. *Annals of Pure and Applied Logic*, pages 61–78, 2006.

[3] Marina Dorzhieva, Rod Downey, Ellen Hammat, Alexander Melnikov, and Keng Meng Ng. Punctually presented structures II: comparing presentations. Submitted.

[4] Rod Downey, Noam Greenberg, Alexander Melnikov, Keng Meng Ng, and Dan Turetsky. Punctual categoricity and universality. *Journal of Symbolic Logic*, 85:1427–1466, 2020.

[5] Rod Downey, Alexander Melnikov, and Keng Meng Ng. On Δ^0_2-categoricity of equivalence relations. *Annals of Pure and Applied Logic*, 166:851–880, 2015.

[6] Noam Greenberg, Matthew Harrison-Trainor, Alexander Melnikov, and Dan Turetsky. Non-density in punctual computability. *Annals of Pure and Applied Logic*, 172:102985, 2021.

[7] Asher M. Kach and Daniel Turetsky. Δ^0_2-categoricity of equivalence structures. *New Zealand Journal of Mathematics*, 39:143–149, 2009.

[8] Iskander Kalimullin, Alexander Melnikov, and Keng Meng Ng. Algebraic structures computable without delay. *Theoretical Computer Science*, 674:73–98, 2017.

[9] Iskander Kalimullin, Alexander Melnikov, and Keng Meng Ng. The diversity of categoricity without delay. *Algebra and Logic*, 56:171–177, 2017.

[10] Iskander Kalimullin, Alexander Melnikov, and Maxim Zubkov. Punctual degrees and lattice embeddings. *Aspects of Computation and Automata Theory with Applications*, pages 315–334, 2023.

[11] Heer Tern Koh, Alexander Melnikov, and Keng Meng Ng. A non-density aspect of the rationals. Submitted.

[12] Anatoly I. Mal'tsev. Constructive algebras I. *Russian Mathematical Surveys*, 16:77–129, 1961.

[13] Alexander Melnikov and Keng Meng Ng. The back-and-forth method and computability without delay. *Israel Journal of Mathematics*, 234:959–1000, 2019.

[14] Alexander Melnikov and Keng Meng Ng. A structure of punctual dimension two. *Proceedings of the American Mathematical Society*, 148:3113–3128, 2020.

[15] Michael O. Rabin. Computable algebra, general theory and theory of computable fields. *Transactions of the American Mathematical Society*, 95:341–360, 1960.

School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore

School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore