The \( \Pi_3 \)-theory of the \( \Sigma_2^0 \)-enumeration degrees is Undecidable

Thomas F. Kent∗

September 22, 2005

Abstract

We show that in the language of \( \{ \leq \} \), the \( \Pi_3 \)-fragment of the first order theory of the \( \Sigma_2^0 \)-enumeration degrees is undecidable. We then extend this result to show that the \( \Pi_3 \)-theory of any substructure of the enumeration degrees which contains the \( \Delta_0^2 \)-degrees is undecidable.

1 Introduction

Intuitively, we say that a set \( A \) is enumeration reducible to a set \( B \) if there is an effective procedure to enumerate \( A \) given any enumeration of \( B \). More formally, given a computably enumerable functional \( \Phi \), we define

\[
\Phi^B = \{ x : \exists (x, F) \in \Phi \land F \subseteq B \land F \text{ is finite} \}
\]

where we identify the finite set \( F \) with a natural number (its canonical index) and \( \langle \cdot, \cdot \rangle \) is a computable bijection from pairs of natural numbers to natural numbers.

We say that \( A \) is enumeration reducible to \( B \), \( A \leq_e B \), if there is a computably enumerable functional \( \Phi \) such that \( A = \Phi^B \). The relation \( \leq_e \) is a pre-order on the powerset of natural numbers and, as such, generates an equivalence relation, denoted \( \equiv_e \), on the powerset of the natural numbers. By \( \text{deg}_e(A) \), we denote the equivalence class, or enumeration degree, of the set \( A \). The least enumeration degree, \( 0_e \), is the set of c.e. sets since trivially, \( A \leq_e 0 \) for every c.e. set \( A \). The enumeration degrees form an upper semi-lattice where we define \( a \lor b = \text{deg}_e(a \oplus b) \) with \( A \in a \) and \( B \in b \). Case [Cas71] proved that the enumeration degrees do not form a lattice by showing that every countable non-principal ideal has an exact pair.

McEvoy [McE85] defined a jump operation on the enumeration degrees that was later expanded by Cooper [Coo84]. Under this definition, we have that \( 0_e' = \text{deg}_e(K) \), the enumeration degree of the compliment of the halting problem. Cooper [Coo84] showed that the degrees below \( 0_e' \) coincide with the set of degrees of \( \Sigma_2^0 \)-sets and every such degree contains only \( \Sigma_2^0 \)-sets. In the same paper, Cooper proved that \( \Sigma_2^0 \)-degrees are dense (see also [LS92]).

Ahmad [Ahm89] showed further lattice theoretic results about the \( \Sigma_2^0 \)-degrees by proving that the diamond lattice embeds into the \( \Sigma_2^0 \)-enumeration degrees preserving 0 and 1 (cf.

∗The author would like to thank his doctoral advisor, Steffen Lempp, whose guidance made this paper possible.
[Ahm91]) and that there are non-splitting \( \Sigma^0_2 \)-enumeration degrees (cf. [AL98]). These results stand in sharp contrast to Lachlan’s [Lac66] Non-Diamond Theorem, and Sacks’ [Sac63] Splitting Theorem for the c.e. Turing degrees. By extending Ahmad’s Diamond Theorem, Lempp and Sorbi [LS02] proved that every finite lattice is embeddable into the \( \Sigma^0_2 \)-enumeration degrees preserving 0 and 1.

In addition to questions about the algebraic structure of the \( \Sigma^0_2 \)-enumeration degrees, it is interesting to ask questions about the decidability of the first order theory of this structure, in the language of \( \{\leq\} \). For example, since there exists a countable independent antichain of degrees, it follows that any finite partial order can be embedded into the \( \Sigma^0_2 \)-enumeration degrees. (The same result could be obtained by applying the lattice embedding theorem of Lempp and Sorbi [LS02].) Since a \( \Sigma_1 \)-sentence describes a finite partial order, it follows that a \( \Sigma_1 \)-sentence is true if and only if it describes a consistent partial order.

On the side of undecidability, Slaman and Woodin [SW97] showed that the \( \Pi_3 \)-fragment of the first order theory of the \( \Sigma^0_2 \)-enumeration degrees is undecidable, by embedding the class of finite graphs into the \( \Sigma^0_2 \)-enumeration degrees, using parameters.

We improve on the result of Slaman and Woodin by showing that the \( \Pi_3 \)-theory of the \( \Sigma^0_2 \)-enumeration degrees is undecidable. We do this by showing that every finite bi-partite graph can be effectively embedded into this structure in a uniform way using parameters. The construction is performed in such a way that it is also shown that the \( \Pi_3 \)-theory of any substructure of the enumeration degrees which contains the \( \Delta^0_4 \)-enumeration degrees is undecidable.

This leaves open the question as to whether the \( \Pi_2 \)-theory is decidable. The question of the decidability of the \( \Pi_2 \)-theory can be rephrased in purely algebraic terms as follows.

**Question 1.1.** Is it possible to effectively decide if, given finite posets \( P \subseteq Q_0, \ldots, Q_n \) for some \( n \geq 0 \), any embedding of \( P \) into the \( \Sigma^0_2 \)-enumeration degrees can be extended to the embedding of some \( Q_i \)? (The choice of \( i \) may depend on the embedding of \( P \).)

Recently, Lempp, Slaman, and Sorbi [LSS], showed that the case when \( n = 0 \) is decidable. This case is known as the Extensions of Embeddings problem. The cases for \( n > 0 \) are still open.

2 Theorems and the algebraic component of the proof

This section and the next closely follow Lempp, Nies and Slaman [LNS98]. Our main result is

**Theorem 2.1.** The \( \Pi_3 \)-theory of the \( \Sigma^0_2 \)-enumeration degrees in the language of partial orderings is undecidable.

We recall that a set of first order sentences \( S \) is *hereditarily undecidable* if there is no computable set of sentences separating \( S \) and \( S \cap V \) where \( V \) is the set of all valid sentences in the language of \( S \). The proof of Theorem 2.1 uses the following theorem:

**Theorem 2.2.** [Nie96] The \( \Sigma_2 \)- (and hence the \( \Pi_3 \)-) theory of the finite bipartite graphs with nonempty left and right domains in the language of one binary relation, but without equality, is hereditarily undecidable.
We will use Theorem 2.2 to prove Theorem 2.1 via the Nies Transfer Lemma. Before we state this lemma, we need to define what it means for one class of structures to be elementarily definable in another class of structures.

**Definition 2.3.** A $\Sigma_k$-formula is a prenex formula that begins with an $\exists$-quantifier and contains $k-1$ quantifier alternations. A $\Pi_k$-formula is a prenex formula that begins with a $\forall$-quantifier and contains $k-1$ quantifier alternations.

**Definition 2.4.** Let $\mathcal{L}_C$ and $\mathcal{L}_D$ be finite relational languages not necessarily containing equality.

1. A $\Sigma_k$-scheme $s$ for $\mathcal{L}_C$ and $\mathcal{L}_D$ consists of a $\Sigma_k$-formula $\varphi_U(\bar{x}; \bar{y})$ (in the language $\mathcal{L}_D$), and for each $m$-ary relation symbol $R \in \mathcal{L}_C$, two $\Sigma_k$-formulas $\varphi_R(\bar{x}_0, \ldots, \bar{x}_{m-1}; \bar{y})$ and $\varphi_{\sim R}(\bar{x}_0, \ldots, \bar{x}_{m-1}; \bar{y})$ (again in $\mathcal{L}_D$).

2. For a $\Sigma_k$-scheme $s$, we define a $\Pi_{k+1}$-formula $\alpha(\bar{p})$, called a correctness condition, for a list of parameters $\bar{p}$, as the conjunction of the following formulas:

   (a) (coding the universe) $\{ \bar{x} : \varphi_U(\bar{x}, \bar{p}) \} \neq \emptyset$, and

   (b) (coding the relations) for each $m$-ary relation symbol $R$ in the language $\mathcal{L}_C$, the set
   \[
   \{(\bar{x}_0, \ldots, \bar{x}_{m-1}) : \forall i < m(\varphi_U(\bar{x}_i, \bar{p}))\}
   \]
   and
   \[
   \{(\bar{x}_0, \ldots, \bar{x}_{m-1}) : \varphi_R(\bar{x}_0, \ldots, \bar{x}_{m-1}, \bar{y})\}.
   \]

3. Define a formula $\varphi_{eq(C)}(x, y)$ as the conjunction of all formulas $\forall \bar{z}(R(x, \bar{z}) \leftrightarrow R(y, \bar{z}))$ where $R$ ranges over all relations $R \in \mathcal{L}_C$ and over all permutations of the arguments of $R$. (The purpose of this formula is to redefine equality if the language contains equality.) For an $\mathcal{L}_C$-structure $C$, define the induced quotient structure $C/eq(C)$ in the obvious way. Similarly define a formula $\varphi_{eq(D)}(x, y)$ and a quotient structure $D/eq(D)$, using the relations $R \in \mathcal{L}_D$.

4. A class $C$ of relational structures, in the language $\mathcal{L}_C$, is $\Sigma_k$-elementarily definable with parameters in a class of relational structures $D$, in the language $\mathcal{L}_D$, if there is a $\Sigma_k$-scheme $s$ such that for each structure $C \in \mathcal{C}$, there is a structure $D \in \mathcal{D}$ and a finite set of parameters $\bar{p} \in D$ satisfying the following:

   (a) (correctness condition) $D \models \alpha(\bar{p})$, and

   (b) (coding the structure) $C/eq(C) \cong \hat{C}/eq(\hat{C})$, where $\hat{C}$ is the substructure of $D$ defined by $\hat{C} = \{ \bar{x} : \varphi_U(\bar{x}, \bar{p}) \}$, and for each $m$-ary relation symbol $R \in \mathcal{L}_C$, the relation $\hat{R}$ on $\hat{C}$ is defined by
   \[
   \hat{R} = \{(\bar{x}_0, \ldots, \bar{x}_{m-1}) : \varphi_R(\bar{x}_0, \ldots, \bar{x}_{m-1}, \bar{p})\}.
   \]

We state two more theorems that are needed to prove our main result.

**Theorem 2.5** (Nies Transfer Lemma [Nie96]). Fix $k \geq 1$ and $r \geq 2$. Suppose a class of structures $C$ is $\Sigma_k$-elementarily definable with parameters in a class of structures $D$ (in finite relational languages $\mathcal{L}_C$ and $\mathcal{L}_D$, respectively). Then the hereditary undecidability of the $\Pi_{r+1}$-theory of $C$ implies the hereditary undecidability of the $\Pi_{r+k}$-theory of $D$.

**Theorem 2.6.** The class of finite bipartite graphs with nonempty left and right domains in the language of one binary relation, but without equality, is $\Sigma_1$-elementarily definable, with parameters, in the partial ordering of the $\Sigma_2^0$-enumeration degrees (i.e. in the class $\{S\}$).
The balance of this chapter after the current section is dedicated to proving Theorem 2.6. The presented construction considers only the cases when the sizes of the left and right domains are both greater than or equal to two. This is done to simplify the construction but in no way affects the result since the $\Sigma_2$-theory of this subclass of structures is also undecidable. The construction is easily modified to accommodate all finite bipartite graphs with non-empty left and right domains; however, the extra technical details that come with this addition obfuscate the finer points of what is happening.

As a side note, we mention that the method of coding finite bipartite graphs was used in [LN95] to establish the undecidability of the $\Pi_4$-theory of the enumerable wtt-degrees and in [LNS98] to establish the undecidability of the $\Pi_3$-theory of the computably enumerable Turing degrees.

Proof of Theorem 2.1. Apply the Nies Transfer Lemma (setting $k = 1$ by Theorem 2.6 and $r = 2$ by Theorem 2.2) in order to obtain the hereditary undecidability of the $\Pi_3$-theory of the $\Sigma_0^0$-enumeration degrees. \hfill $\Box$

We will perform the construction in such a way as to show the following corollary:

**Corollary 2.7.** The $\Pi_3$-theory of the $\Delta_2^0$-enumeration degrees in the language of partial orderings is undecidable.

*Proof.* The proof of Theorem 2.6 actually shows that the class of finite bipartite graphs with nonempty left and right domains in the language without equality is $\Sigma_1$-elementarily definable in the partial ordering of the $\Sigma_0^0$-enumeration degrees using $\Delta_2^0$-degrees as parameters. Since the $\Delta_2^0$-degrees are a proper subclass of the $\Sigma_2^0$-degrees, we are able to restrict all quantifiers in the defined $\Sigma_1$-scheme to $\Delta_2^0$-degrees. \hfill $\Box$

Once we have shown that both the $\Delta_2^0$- and $\Sigma_2^0$-enumeration degrees are undecidable, by an extension of the above argument, we get the following corollary:

**Corollary 2.8.** If $\mathcal{M}$ is a substructure of the enumeration degrees which contains the $\Delta_2^0$-degrees then the $\Pi_3$-theory of $\mathcal{M}$ is undecidable.

## 3 The requirements

In this section we will introduce the requirements that need to be satisfied to prove the main theorem and justify how their satisfaction implies the desired result. We begin with a definition.

**Definition 3.1.** Let $I$ be a computable subset of $\omega$. We say that the set of degrees $\{a_i : i \in \omega\}$ is independent if for every $j \in I$, $a_j \not\leq \bigvee_{i \in I-\{j\}} a_i$.

Fix a finite bipartite graph with nonempty left domain $L = \{0, 1, \ldots, n\}$, nonempty right domain $R = \{\tilde{0}, \tilde{1}, \ldots, \tilde{n}\}$ and edge relation $E \subseteq L \times R$.

We code the left domain using a $\Sigma_1$-formula $\psi(x; a, b, c)$. We will represent each vertex $i \in L$ by a difference of two intervals $[a_i, a) - [c, 0']$ of $\Sigma_2^0$-degrees ($0'$ is defined later) where the following properties hold:

\begin{align*}
(3.1) \quad & a = \bigvee_{i \in L} a_i; \\
(3.2) \quad & \text{for all } i, j \in L, \text{ if } i \neq j \text{ then } c \leq a_i \lor a_j;
\end{align*}
\[(3.3) \quad \text{for all } i \in L, \ c \not\leq a_i; \]

\[(3.4) \quad \text{the degrees } a_0, \ldots, a_n \text{ are independent; and} \]

\[(3.5) \quad \text{there exists a } \Sigma_2^0 \text{-enumeration degree } b \text{ incomparable with each } a_i \text{ and } a, \text{ such that } \forall x \leq a(x \not\leq b \iff \exists i \leq n(a_i \leq x)). \]

The \( \Sigma_1 \)- (in fact quantifier free) formula \( \psi(x; a, b, c) \) used to code the left domain is now chosen to be
\[
x \leq a \& x \not\leq b \& x \not\geq c. \]

In the course of the construction, we build \( \Sigma_2 \)-sets \( A_0, \ldots, A_n, A, B, \) and \( C, \) and set \( a_i = \deg_e(A_i) \) for all \( i \in L, \) \( a = \deg_e(A), b = \deg_e(B) \) and \( c = \deg_e(C). \) (Even though we build these sets as \( \Sigma_2^0 \)-sets, we will actually construct them using \( \Delta_2^0 \)-approximations.)

We now outline the requirements that these sets need to meet in order to satisfy the above properties.

To ensure (3.1) and (3.2), for all \( i, j \in L \) with \( i \neq j, \) we construct enumeration operators \( \Theta_{i,j} \) to meet the global requirements:
\[
\begin{align*}
\mathcal{J} : A &= \bigoplus_{i \in L} A_i = \text{Definition } \{ \langle x, i \rangle : x \in A_i \}, \\
\mathcal{P}_{i,j} : C &= \Theta_{i,j}^{A_i \oplus A_j} \text{ if } i \neq j.
\end{align*}
\]

We ensure (3.3) and (3.4) by requiring for all enumeration operators \( \Xi \) and \( \Psi, \) and all \( i \in L: \)
\[
\begin{align*}
N_{\Xi,i} : C &\neq \Xi^{A_i} \text{ and} \\
I_{\Psi,j} &\neq A_i \neq \Psi^{\bigoplus_{i \neq j} A_j}.
\end{align*}
\]

Finally, in order to ensure (3.5), we require that for all enumeration operators \( \Phi \) and \( \Omega \) and for all \( j \in L: \)
\[
\begin{align*}
S_{\Omega} : \exists \Gamma(\Omega^A = \Gamma^B) &\text{ or } \exists \Delta, i \in L(A_i = \Delta^{\Omega^A}), \\
T_{\Phi,j} &\neq A_j \neq \Phi^B,
\end{align*}
\]

where \( \Gamma \) and \( \Delta \) are enumeration operators built by us that depend on \( \Omega \) and \( j. \) We mention here that the requirements \( S_{\Omega} \) and \( T_{\Phi,j} \) generalize a theorem of Ahmad. In \([\text{Ahm89}]\) \( (\text{c.f. } [\text{AL98}]) \) she constructs what is known as an Ahmad pair: two \( \Sigma_2 \)-enumeration degrees \( a \) and \( b \) such that \( a \not\leq b \) but for all degrees \( c \not< a, c \not< b. \)

The right domain is coded in a similar manner using \( \Sigma_2 \)-sets \( \hat{A}_0, \ldots, \hat{A}_n, \hat{A}, \hat{B} \) and \( \hat{C}, \) and requirements \( \tilde{J}, \tilde{P}_{i,j}, \tilde{N}_{\Xi,i}, \tilde{I}_{\Psi,j}, \tilde{S}_{\Omega} \) and \( \tilde{T}_{\Phi,j}. \) The \( \Sigma_1 \)-formula \( \varphi_U(x; \hat{y}) \) required by Definition 2.4 can now be chosen as \( \psi(x; a, b, c) \lor \psi(x; \hat{a}, \hat{b}, \hat{c}). \)

The reason that we use an ambiguous representation of the vertices is that we need a \( \Sigma_1 \)-formula to define the universe. We could represent the left domain by the minimal degrees satisfying \( \psi(x; a, b, c), \) i.e. \( \{a_0, \ldots, a_n\}, \) but this would be a \( \Pi_1 \)-formula and hence only imply that the \( \Pi_1 \)-theory of \( \mathfrak{U} \) is undecidable. Given a degree \( x \) that satisfies \( \psi(x; a, b, c), \) properties (3.1) - (3.4) allow us to unambiguously recover the vertex that this degree represents.

In defining a copy of the edge relation \( E(\cdot, \cdot), \) we need to make sure that the formulas \( \varphi_E(x, \hat{x}, \hat{y}) \) and \( \varphi_{\mathfrak{U}}(x, \hat{x}, \hat{y}) \) do not depend on the particular pair of degrees that are chosen to represent the vertices. To accomplish this, we build two more \( \Sigma_2 \)-enumeration degrees \( e_0 \) and \( e_1 \) satisfying for all \( i \in L \) and \( i \in R: \)
(3.6) \( E(i, \tilde{i}) \) iff \( e_0 \leq a_1 \lor \tilde{a}_i \) iff \( e_1 \not\geq a_i \lor \tilde{a}_i \), and

(3.7) \( \neg E(i, \tilde{i}) \) iff \( e_0 \not\geq a_i \lor \tilde{a}_i \) iff \( e_1 \leq a_i \lor \tilde{a}_i \).

The \( \Sigma_1 \)-formula \( \varphi_E(x, \tilde{x}, e_0, e_1) \) required by Definition 2.4 can now be chosen as:

\[
(\exists x_1 \leq x)(\exists \tilde{x}_1 \leq \tilde{x})(\exists z)(\psi(x_1, a, b, c) \land \psi(\tilde{x}_1, \tilde{a}, \tilde{b}, \tilde{c}) \land x_1 \geq z_1 \land \tilde{x}_1 \geq z_1 \land e_1 \not\geq z)
\]

The choice of the \( \Sigma_1 \)-formula for \( \varphi_{-E}(x, \tilde{x}, e_0, e_1) \) is similar (the only difference is that \( e_1 \) has been replaced by \( e_0 \)):

\[
(\exists x_1 \leq x)(\exists \tilde{x}_1 \leq \tilde{x})(\exists z)(\psi(x_1, a, b, c) \land \psi(\tilde{x}_1, \tilde{a}, \tilde{b}, \tilde{c}) \land x_1 \geq z_1 \land \tilde{x}_1 \geq z_1 \land e_0 \not\geq z)
\]

To ensure the equivalences dictated by (3.6) and (3.7), we build \( \Sigma^0_2 \)-sets \( E_0 \) and \( E_1 \), setting \( e_0 = \deg_e(E_0) \) and \( e_1 = \deg_e(E_1) \), and meeting for each enumeration operator \( \Upsilon \), each \( i \in L \), and each \( \tilde{i} \in R \), the following requirements:

\[
E^0_{i, \tilde{i}} : E(i, \tilde{i}) \Rightarrow E_0 = \Lambda^A_i \oplus \tilde{A}_i,
\]

\[
F^1_{i, \tilde{i}} : E(i, \tilde{i}) \Rightarrow E_1 = \tilde{\Upsilon}^A_i \oplus \tilde{A}_i,
\]

\[
E^1_{i, \tilde{i}} : \text{not } E(i, \tilde{i}) \Rightarrow E_1 = \Lambda^A_i \oplus \tilde{A}_i,
\]

\[
F^0_{i, \tilde{i}} : \text{not } E(i, \tilde{i}) \Rightarrow E_0 \not\equiv \tilde{\Upsilon}^A_i \oplus \tilde{A}_i.
\]

where \( \Lambda^A_i \) and \( \tilde{\Lambda}^A_i \) are enumeration operators built by us.

It is clear that the above requirements establish conditions (3.1) - (3.7) and that the formulas \( \varphi_U \), \( \varphi_E \), and \( \varphi_{-E} \) establish Theorem 2.6.

Finally, in order to show Corollary 2.7, we add an additional requirement which when met ensures that \( A \) and \( \tilde{A} \) are low. The following definitions, theorems, and lemma motivate the requirement that we use:

**Definition 3.2.** [Coo87], [McE85] Given a set \( A \subset \omega \), we define

- \( K_A = \{ x : x \in \Phi^A_x \} \), where \( \Phi_x \) is the \( x \)-th enumeration operator under some fixed computable listing;

- the jump of \( A \) to be \( A' = \text{Definition } K_A \oplus \overline{K_A} \); and

- \( a' = \deg_e(A') \) where \( a = \deg_e(A) \).

Cooper and McEvoy show that the jump operator in the enumeration degrees has the same properties as the jump operator in the Turing degrees. Namely, \( A \leq_e B \Rightarrow A' \leq eB' \) and \( A <_e A' \).

**Theorem 3.3.** [McE85] \( 0'_e = \deg_e(\emptyset') = \deg_e(\overline{K}) \) where \( K \) denotes the compliment to the halting problem.

**Theorem 3.4.** [Coo87] \( 0'_e \) is the maximal \( \Sigma^0_2 \)-enumeration degree. i.e. \( A \leq_e \overline{K} \) if and only if \( A \) is \( \Sigma^0_2 \).

**Definition 3.5.** An enumeration degree \( a \) is low if \( a' = 0'_e \). A set is low if its enumeration degree is low.

**Lemma 3.6.** [MC85] A set \( A \) is low if and only if there exists a \( \Delta^0_2 \)-approximation to \( A \) such that for all enumeration operators \( \Phi \) and all \( x \), \( \lim_s \Phi^A_s(x) \) exists.
Thus, satisfying the requirement
\[ L_{\Pi,x} : \exists^\omega s \left( x \in \Pi^{A \oplus \tilde{A}}[s] \right) \Rightarrow x \in \Pi^{A \oplus \tilde{A}} \]
for all enumeration operators \( \Pi \) and all \( x \in \omega \) will guarantee that \( A \) and \( \tilde{A} \) are both low, as well as \( A_i \) and \( \tilde{A}_i \) for all \( i \in L \) and \( \tilde{i} \in R \). We construct \( B \) and \( \tilde{B} \) in such a way that the lowness of \( A \) and \( \tilde{A} \) guarantees that \( B, \tilde{B}, C, \tilde{C}, E_0, \) and \( E_1 \) are \( \Delta^0_2 \).

4 The intuition for the strategies

We briefly outline the strategies used to meet the above requirements. The key part of the construction, and the part that makes it a \( 0^{\prime\prime\prime} \) construction, is the interplay between the \( S \)- and the \( T \)-strategies. We will first explain this interaction and then add the \( J \)-, \( P \)-, \( N \)- and \( I \)-strategies. In describing the interaction between the \( S \)- and \( T \)-strategies, their action in the actual construction, and in their verification, we closely follow the construction of Lempp, Slaman, and Sorbi [LSS]. Since the strategies for the left and the right domains are the same, in what follows, we initially only describe the strategies that are necessary for building the left domain. We then add the \( E^j \)-, \( F^j \)-, and \( L \)-strategies, which are the strategies that define the relationship between the left and right domains.

The \( S \)-requirement

This strategy will try to build \( \Gamma \) while lower priority \( T \)-requirements try to destroy \( \Gamma \) and build \( \Delta \). The strategy is as follows:

1. Pick the least element \( x \in \Omega^A \) that has no coding number \( b_x \).
2. Pick a coding number \( b_x \) for \( x \) larger than any number seen so far in the construction.
3. Enumerate \( b_x \) into \( B \) and the axiom \( \langle x, F^x \rangle \) into \( \Gamma \) where \( F^x \) is the finite set composed of \( b_x \) and all current killing points for this strategy (picked by lower priority \( T \)-requirements).
4. For all \( x \in \Omega^A - \Gamma^B \), with \( b_x \) defined, enumerate \( b_x \) into \( B \).
5. For all \( x \in \Gamma^B - \Omega^A \), extract \( b_x \) from \( B \).

Without interference from lower priority \( T \)-requirements, it is clear that \( S \) successfully builds \( \Gamma \).

The \( T \)-requirement

The strategy of the \( T \)-requirement varies markedly depending on whether or not there is an active \( S \)-requirement above it on the tree of strategies. Hence, we will slowly work up to the full strategy (action below several active higher priority \( S \)-requirements) in three stages.

One \( T \)-requirement in isolation

In isolation, the \( T \)-requirement follows the basic Friedberg-Muchnik strategy as follows: A witness \( a \) is chosen from a stream (defined below) of available witnesses and enumerated into the set \( A_j \). When, if ever, the element \( a \) enters \( \Phi^B \), the strategy will extract \( a \) from \( A_j \) while restraining \( B \). Since there is no active higher priority \( S \)-requirement, we do not have to worry about a \( B \)-correction, in response to this extraction, that may injure our computation.
One $T$-requirement below one $S$-requirement

The case where one $T$-strategy is below one $S$-strategy is somewhat more complicated. The strategy proceeds as above and tries to find a number $a \in \Phi^B$ which can be extracted from $A_j$ while still maintaining $\Gamma^B \subseteq \Omega^A$ and $a \in \Phi^B$. If such a number is ever found, the strategy will diagonalize and stop. If no such number is found, a stream of elements will be generated such that the removal of any one of these elements from $A_j$ causes numbers to leave $B$. We then restrict all future changes of $A_j$ to elements from this stream. This will put us in a position to meet the $S$-requirement via the second alternative, at the expense of failing to achieve the $T$-requirement, by destroying $\Gamma$ and building a $\Delta$ which allows us to calculate $\Phi^A$ from $B$.

More precisely, the $T$-strategy proceeds as follows:

1. Pick a fresh killing point $q$ for $\Gamma$. Put $q$ into $B$ and require all future $\Gamma$-axioms $\langle x, F_x \rangle$ to include $q$ in the oracle set $F_x$.

2. Pick a fresh witness $z$ and put $z$ into $A_j$.

3. Wait for $z \in \Phi^B$ via some axiom $\langle z, F \rangle$ at some future stage $s$.

4. Extract $z$ from $A_j$ and allow the $S$-strategy to correct $B$ (possibly injuring $\Phi^B(z)$).

5. From now on, if ever $\Gamma^B[s] \subseteq \Phi^A$ (while $z \notin A_j$), then cancel all action between stage $s$ and now, restrain $F \subseteq B$, and stop. (In this case, we call the computation $\Phi^B(z)$ $\Gamma$-cleared.)

6. While waiting for Step 5 to apply, put $z$ into the stream $S$; restrict all future changes in $A_j \restriction s$ to numbers in $S$; extract $q$ from $B$; add the axiom $\langle z, \Gamma^B[s] \rangle$ in $\Delta$; add axioms $\langle z', \emptyset \rangle$ to $\Delta$ for all $z' < s$ with $z' \in A_j[s] - S$; and restart at Step 1 with a fresh killing point $q$.

The possible outcomes of the above $T$-strategy are as follows:

(A) Wait forever at Step 3: Then $z \in A_j - \Phi^B$, and $\Gamma$ is not affected since $q \in B$.

(B) Stop eventually at Step 5: Then $z \in \Phi^B - A_j$, and $\Gamma$ is not affected since $z \in B$.

(C) Looping between Step 1 and Step 6 infinitely often: Then the $T$-requirement may not be satisfied by the action of this strategy. Additionally, $\Gamma^B$ will be finite since all killing points are eventually extracted from $B$, and all but finitely many $\Gamma$-axioms $\langle x, F \rangle$ contain one of these killing points in $F$. However, $A_j = \Delta^{\Omega^A}$ can be seen to hold as follows: For all $z \notin S$, the $A_j$-restraint from Step 6 guarantees that $z \in A_j$ if and only if $z \in \Delta^{\Omega^A}$, so we restrict our attention to elements $z \in S$. If $z \in A_j$ (and was enumerated into $S$ at stage $s_z$) then $\Omega^A[s_z] \subseteq \Omega^A$ (assuming that no other strategies remove numbers in $A[s_z]$ from $A$, and so no number from $\Omega^A[s_z]$, the set $\Omega^A$ measured immediately before the extraction of $z$ from $A_j$, can leave $\Omega^A$), implying that $z \in \Delta^{\Omega^A}$. Conversely, if $z \in \Delta^{\Omega^A}$, then $\Gamma^B[s_z] \subseteq \Omega^A$, and since Step 5 never applies, we must have that $z \in A_j$. 

8
One $T$-requirement below several $S$-requirements

In this case, the $T$-strategy is basically a nested version of the previous strategy: If we generate only a finite number of witnesses, or if we find a witness which is $\Gamma$-cleared for all $\Gamma$’s above, then we diagonalize finitarily. Otherwise, we find the lowest priority $S$-requirement such that infinitely many witnesses are not $\Gamma$-cleared for its $\Gamma$. We will then use these witnesses to form a stream from which lower priority strategies will have to work.

In addition to the finitary outcomes, we now have $i_0$ many infinitary outcomes (where $i_0$ is the number of $\Gamma$’s that our strategy has to deal with). More details on this interaction will be given in the formal construction.

One $T$-requirement below another $T$-requirement

Assume that we have one $T$-requirement $G$ below another $T$-requirement $\hat{G}$. If $G$ assumes finite outcome (A) or (B) for $\hat{G}$, then $G$ will act as described above. Otherwise, $G$ assumes the infinite outcome (C) of $\hat{G}$. In this case, $G$ assumes that $\Gamma^B$ is finite and, in fact, will only be able to act at stages in which $\hat{G}$ has extracted the latest killing point $q$ from $B$. Thus, $G$ can now act as if in isolation, the restriction being that it can only use witnesses in the stream defined by $\hat{G}$, so as to keep $\Delta$ correct. Note that when $G$ puts a number $z$ into, or extracts a number $z$ from, the set $A_j$ at a stage $s$, all numbers greater than $z$ are removed from the stream and dumped into $A_i$ since their assumption about $\Omega^A$ may now be incorrect. When a number $z$ is dumped into a set $Z$, $z$ is permanently enumerated into $Z$, and for any functional $\Delta$ that is being built, with $Z = \Delta^X$, the axiom $(z, \emptyset)$ is enumerated into $\Delta$.

The $J$-requirement

The $J$-strategy is a global strategy and operates by defining $(x, i) \in A$ if and only if $x \in A_i$.

The $N$- and $P$-requirements

The $N$-strategy is a standard Friedberg-Muchnik strategy and acts like the $T$-strategy in isolation. The strategy will choose a new coding number $c$ from the stream and enumerate $c$ into $C$. When, if ever, the element $c$ enters $\Xi^A_i$, the strategy will extract $c$ from $C$ while restraining $A_i$.

The $P$-strategy is a global strategy and works in conjunction with the $N$-strategies. Whenever some $N_{\Xi^A_i}$-strategy enumerates an element $c$ into $C$, the $P$-strategy chooses coding numbers $a_j$ (for $j \in L$) from the stream, enumerates $a_j$ into $A_j$, and enumerates the axiom $(c, \{a_j\} \oplus \{a_k\})$ into $\Theta_{j,k}$ (for $j \neq k$). If ever $c$ is extracted from $C$, then $a_j$ is extracted from $A_j$ for all $j \neq i$.

The $I$-requirements

The $I$-requirement is a standard Friedberg-Muchnik strategy and acts like the $T$-strategy in isolation. The strategy will enumerate a new coding number $a$, chosen from the stream of available witnesses, into the set $A_i$. When, if ever, the element $a$ enters $\Psi^{\oplus_j \neq i, A_i}$, the strategy will extract $a$ from $A_i$ while restraining $\bigoplus_{j \neq i} A_j$. 

9
The $\mathcal{E}^0$-, $\mathcal{F}^0$-, $\mathcal{F}^1$-, and $\mathcal{E}^1$-requirements

In this section, we describe the action of the strategies that define the edge relationship between the vertices in $L$ and those in $R$. For the same reason that we only described the strategies that deal with the left domain, here we will only describe the action of the $\mathcal{F}^0$- and $\mathcal{E}^0$-strategies in that the $\mathcal{F}^1$- and $\mathcal{E}^1$-strategies have the same behavior.

The $\mathcal{F}^0$-strategy

We assume that $i \in L$, $i \in R$, and $\neg E(i, i)$. Like the $\mathcal{T}$-requirement, this is a standard Friedberg-Muchnik strategy and acts just like a $\mathcal{T}$-strategy in isolation. The strategy will enumerate a new coding number $e$, larger than any number seen so far in the construction, and not from the stream, into the set $E_0$. When, if ever, the element $e$ enters $\Upsilon^{A_i \oplus \hat{A}_i}$, the strategy extracts $e$ from $E_0$ while restraining $A_i \oplus \hat{A}_i$.

The $\mathcal{E}^0$-strategy

This is a global strategy which works in conjunction with the $\mathcal{F}^0$-strategies, and builds an enumeration functional $\Xi_{0,i,j}$ for every $i \in L$ and $i \in R$ with $E(i, i)$. Whenever an element $e$ is enumerated into $E_0$, for every $i \in L$ and $i \in R$, new coding numbers $a_i$ and $\tilde{a}_i$ are chosen from the stream, and the axiom $\langle e, \{a_i\} \oplus \{\tilde{a}_i\} \rangle$ is enumerated into $\Xi_{0,i,j}$. If $e$ is ever extracted from $E_0$ by some $\mathcal{F}_{T,j}^0$-strategy, the $\mathcal{E}^0$-strategy extracts those $a_i$ from $A_i$ and $\tilde{a}_i$ from $\hat{A}_i$ with $i \neq j$ and $i \neq \hat{j}$.

The $\mathcal{L}$-strategy

The action of the lowness strategy is similar to the $\mathcal{N}$-strategy, however it picks no coding numbers and only restraints $A \oplus A$. Specifically, the strategy waits for $x$ to enter $\Phi^{A \oplus \hat{A}}$ and when, if ever, this happens, restraints $A \oplus \hat{A}$ by restraining all set $A_i$ and $\hat{A}_i$ for $i \in L$ and $i \in R$.

5 The tree of strategies

For the sake of simplifying notation, in what follows we will refer to the $\mathcal{F}^0$- and $\mathcal{F}^1$-requirements as $\mathcal{F}$-requirements, the $\mathcal{I}$- and $\mathcal{I}$-requirements as $\mathcal{I}$-requirements, the $\mathcal{S}$- and $\mathcal{S}$-requirements as $\mathcal{S}$-requirements, etc. Fix an arbitrary effective priority ordering $\{R_e\}_{e \in \omega}$ of all $\mathcal{N}$-, $\mathcal{T}$-, $\mathcal{S}$-, $\mathcal{T}$-, $\mathcal{F}$-, and $\mathcal{L}$-requirements. The $\mathcal{F}$-, $\mathcal{P}$- and $\mathcal{E}$-requirements will not be put on the tree of strategies since they are handled globally. Furthermore, we only put an $\mathcal{F}$-requirement into the priority ordering if its assumption about the edge relationship is true.

We define $\Sigma = \{\text{stop} < \infty_0 < \infty_1 < \infty_2 < \cdots < \text{wait} < \text{so}\}$ as our set of outcomes. (“so” stands for “$\mathcal{S}$‘s outcome”.) We define $\mathcal{T} \subset \Sigma^{\leq \omega}$ and refer to it as our tree of strategies. Each node of $\mathcal{T}$ will be associated with, and thus identified with, a strategy.

We assign requirements to nodes on $\mathcal{T}$ by induction as follows: The empty node is assigned to requirement $R_0$, and no requirement is active or satisfied along the empty node. Given an assignment to a node $\alpha \in \mathcal{T}$, we distinguish cases depending on the requirement $R$ assigned to $\alpha$:

Case 1: $R$ is an $\mathcal{S}$-requirement: Then call $R$ active along $\alpha \sim \langle \text{so} \rangle$ via $\alpha$. For all other requirements $R'$, call $R'$ active or satisfied along $\alpha \sim \langle \text{so} \rangle$ via $\beta \subset \alpha$ if and only if it is so.
along $\alpha$. Assign to $\alpha^\prec\langle o \rangle$ the highest priority requirement that is neither active nor satisfied along $\alpha^\prec\langle o \rangle$.

Case 2: $R$ is an $\mathcal{N}$-, $\mathcal{I}$-, $\mathcal{F}$-, or $\mathcal{L}$-requirement. Then for $o \in \{\text{stop, wait}\}$, call $R$ satisfied along $\alpha^\prec\langle o \rangle$ via $\alpha$; and for all other requirements $R'$, call $R'$ active or satisfied along $\alpha^\prec\langle o \rangle$ via $\beta \subset \alpha$ if and only if it is so along $\alpha$. Assign to $\alpha^\prec\langle o \rangle$ (for $o \in \{\text{stop, wait}\}$) the highest priority requirement that is neither active nor satisfied along $\alpha^\prec\langle \text{wait} \rangle$.

Case 3: $R$ is a $T$-requirement. Let $\beta_0 \subset \cdots \subset \beta_{i_0-1}$ be all the strategies $\beta \subset \alpha$ such that some $\mathcal{S}$-requirement is active along $\alpha$ via $\beta_i$. We denote by $S_i$ the $\mathcal{S}$-requirement for $\beta_i$. (Here we allow $i_0 = 0$, in which circumstance this case is handled the same way as Case 2.) Then, for $o \in \{\text{stop, wait}\}$, call $R$ satisfied along $\alpha^\prec\langle o \rangle$ via $\alpha$; and for all other requirements $R'$, call $R'$ active or satisfied along $\alpha^\prec\langle o \rangle$ via $\beta \subset \alpha$ if and only if it is so along $\alpha$. If $i_0 > 0$, fix $i \in [0, i_0)$. Call $S_i$ satisfied along $\alpha^\prec\langle \infty_i \rangle$ via $\beta_i$ and call any $S_j$ requirement, for $j \in \langle i, i_0 \rangle$, neither active nor satisfied along $\alpha^\prec\langle \infty_i \rangle$; any other requirement is active or satisfied along $\alpha^\prec\langle \infty_i \rangle$ via $\beta \subset \alpha$ if and only if it is so along $\alpha$. For any outcome $o \in \{\text{stop, wait}\} \cup \{\infty_i : i \in [0, i_0)\}$, assign to $\alpha^\prec\langle o \rangle$ the highest priority requirement neither active nor satisfied along $\alpha^\prec\langle o \rangle$. (The intuition is that under the finitary outcomes (stop) and (wait), the $T$-requirement is assumed to be satisfied finitarily by diagonalization; whereas under outcome $\langle \infty_i \rangle$, the $S_i$-requirement, while previously satisfied via an enumeration operator $\Gamma_i$, is now assumed to be satisfied by $\alpha$ constructing an enumeration operator $\Delta_i$, while all $S_j$-requirements active via some strategy between $\beta_i$ and $\alpha$ are assumed to be injured.)

The tree of strategies $T$ is now the set of all nodes $\alpha \in \Sigma^{<\omega}$ to which requirements have been assigned.

### Notation and terminology for strings

We use the standard notation and terminology of strings which can be found in [Soa87]. In particular, given strings $\alpha$ and $\beta$, we use $\alpha \subseteq \beta$ ($\alpha \subset \beta$) to denote that $\beta$ extends (properly extends) $\alpha$. We say $\alpha$ is to the left of $\beta$ ($\alpha <_L \beta$) if $\alpha$ is lexicographically less than $\beta$ but $\alpha \not\subseteq \beta$. Furthermore, by $\alpha \leq \beta$ we denote non-strict lexicographical ordering ($\alpha <_L \beta$ or $\alpha \subseteq \beta$), and by $\alpha < \beta$ we denote strict lexicographical ordering ($\alpha \leq \beta$ and $\alpha \neq \beta$).

### 6 The construction

The construction proceeds in stages $s \in \omega$. Before beginning, we give some conventions and definitions.

When we initialize a strategy, we make all the parameters undefined and make the stream $S(\alpha)$ of $\alpha$ empty.

The stream $S(\emptyset)$ of the root node $\emptyset$ of our tree of strategies at any stage $s$ is $[0, s)$. The streams $S(\alpha)$ for $\alpha \neq \emptyset$ are defined during the construction.

A strategy will be eligible to act if it is along the current approximation $f_s \in T$ to the true path $f \in [T]$ of the construction. At a stage $s$, if $\alpha \subseteq f_s$, $s$ is called an $\alpha$-stage.

At an $\alpha$-stage $s$, a number $z$ in the stream $S(\alpha)$ is suitable for $\alpha$ if, for every set $X$ in $\{A, A_0, \ldots, A_n, \bar{A}, \bar{A}_0, \ldots, \bar{A}_n\}$,

1. $z$ is not currently in use for $X$ by any strategy (i.e. $z$ is not the current witness or coding number targeted for $X$ by any strategy that has not been initialized since $z$ has been picked).

2. $z$ has not been dumped into $X$. 

11
3. $z$ is greater than $|\alpha|$ or any stage at which any $\beta \supseteq \alpha$ has changed any set, picked any number, or extended any enumeration operator.

4. $z$ is greater than any stage which any $\beta \subset \alpha$ with finitary outcome $\langle \text{wait} \rangle$ or $\langle \text{stop} \rangle$ along $\alpha$ has first taken on this outcome since its last initialization.

5. $z$ is greater than $z'$ many numbers in $S$ which are not in use for $X$ by any $\beta \subseteq \alpha$ where $z'$ is the greater of the last number in use by $\alpha$ and the most recent stage at which $\alpha$ was initialized.

**Definition 6.1.** Given a $\Sigma^0_2$-approximation $\langle X_s \rangle_{s \in \omega}$ to a set $X$, an element $x$, and a stage $s$, we define $a(X; x, s)$, the *age* of $x$ in $X$ at stage $s$, to be the least stage $s_x \leq s + 1$ such that for all stages $t$, if $s_x \leq t \leq s$ then $x \in X_t$. If $Z$ is a finite set, then we define $a(X; Z, s)$, the *age* of $Z$ in $X$ at the stage $s$, to be max $\{a(X; z, s) : z \in Z\}$. Given $\Sigma^0_2$-approximations $\langle X_s \rangle_{s \in \omega}$ and $\langle Y_s \rangle_{s \in \omega}$ to sets $X$ and $Y$ respectively, the *least oldest* element in $X - Y$ at the stage $s$ is the least element $x \in X_s - Y_s$ such that for all $y \in X_s - Y_s$, $a(X, x, s) \leq a(X, y, s)$, and the *least oldest* subset of $X - Y$ at the stage $s$ is the least $F \subseteq X_s - Y_s$ such that for all $G \subset X_s - Y_s$, $a(X, F, s) \leq a(X, G, s)$.

During the course of the construction, all parameters are assumed to remain unchanged unless specified otherwise.

At the end of each stage $s$, we will dump certain elements into their respective target sets and initialize certain strategies as described below under *Ending the stage $s$*.

We now proceed with the construction.

*Stage 0:* Initialize all $\alpha \in T$.

*Stage $s > 0$:* Each stage $s$ is composed of substages $t \leq s$ such that some strategy $\alpha \in T$, with $|\alpha| = t$, acts at substage $t$ of stage $s$ and decides which strategy will act at substage $t + 1$ or whether to end the stage. If during a substage, there are no suitable numbers in the stream for that strategy, we end the current stage and continue with stage $s + 1$. The longest strategy eligible to act during a stage $s$ is called the current approximation to the true path at stage $s$ and is denoted $f_s$.

*Substage $t$ of stage $s$:* Suppose a strategy $\alpha$ of length $t$ is eligible to act at this substage. We distinguish cases depending on the requirement $R$ assigned to $\alpha$.

**Case 1:** $R$ is an $S_\Omega$-requirement: For the least oldest $z \in \Omega^A - \Gamma^B$ choose a new coding number $b_{z'}$, if it is not already defined, larger than any number seen so far in the construction. Enumerate $b_t$ into $B$ and the axiom $\langle z, F \rangle$ into $\Gamma$, where the oracle set $F$ contains $b_t$ and all the current killing points $q$ for $\Gamma$ defined by $T$-strategies $\beta \supseteq \alpha^{-}\langle \text{so} \rangle$. For any $z' \in \Gamma^B - \Omega^A$, remove $b_{z'}$ from $B$. End the substage by letting $\alpha^{-}\langle \text{so} \rangle$ be eligible to act next and set the stream $S(\alpha^{-}\langle \text{so} \rangle) = S(\alpha) \cap [s_0, s)$, where $s_0$ is the most recent stage less than or equal to $s$ at which $\alpha$ was initialized.

**Case 2:** $R$ is an $L_{\Pi^A_0}$-requirement: Pick the first subcase which applies:

**Case 2.1:** $x \notin \Pi^{A_{\bar{B}}}$: Let $\alpha^{-}\langle \text{wait} \rangle$ be eligible to act next and set the stream $S(\alpha^{-}\langle \text{wait} \rangle) = S(\alpha) \cap [s_0, s)$, where $s_0$ is the most recent stage $\leq s$ at which $\alpha$ was initialized.

**Case 2.2:** $x \in \Pi^{A_{\bar{B}}}$: Let $\alpha^{-}\langle \text{stop} \rangle$ be eligible to act next and set the stream $S(\alpha^{-}\langle \text{stop} \rangle) = S(\alpha) \cap [s_0, s)$, where $s_0$ is the greater of the most recent stage $\leq s$ when $\alpha$ was initialized and the least stage $s' \leq s$ such that $\alpha$ was active and $x \in \Pi^{A_{\bar{B}}} [s_1]$ for all $s' \leq s_1 \leq s$.

**Case 3:** $R$ is an $N_{\Xi^r}$-requirement: Pick the first subcase which applies.
Case 3.1: $\alpha$ has not been eligible to act since its most recent initialization or some coding number $a_j$, for $j \in L$, is not defined: For each $j \in L$ with $a_j$ undefined, choose a new distinct coding number $a_j$ that is suitable for $\alpha$ and end the current stage.

Case 3.2: All coding numbers $a_j$, for $j \in L$, are defined but the coding number $c$ is not defined: Choose $c$ larger than any number seen so far in the construction. Enumerate $c$ into $C$, $a_j$ into $A_j$ for all $j \in L$, the axioms $\langle c, \{a_j\} \oplus \{a_k\} \rangle$ into $\Theta_{j,k}$ for all $j, k$ with $j \neq k$, and end the current stage.

Case 3.3: The coding number $c$ is defined and $c \notin \Xi^{A_j}$: Let $\alpha \prec \langle \text{wait} \rangle$ be eligible to act next and set the stream $S(\alpha \prec \langle \text{wait} \rangle) = [s_0, s)$ where $s_0$ is the stage at which $c$ was chosen.

Case 3.4: The coding number $c$ is defined and $c \in C \cap \Xi^{A_j}$: Then $\alpha$ stops the strategy by extracting $c$ from $C$, all $a_j$ from $A_j$ with $(a_j \neq a_i)$, and ending the current stage.

Case 3.5: The coding number $c$ is defined and $c \in \Xi^{A_j} - C$: Let $\alpha \prec \langle \text{stop} \rangle$ be eligible to act next and set the stream $S(\alpha \prec \langle \text{stop} \rangle) = [s_0, s)$, where $s_0$ is the stage at which $\alpha$ stopped.

Case 4: $R$ is an $\mathcal{I}_{\Psi,j}$-requirement: Pick the first subcase which applies.

Case 4.1: $\alpha$ has not been eligible to act since its most recent initialization or the coding number $a_i$ is undefined: Choose a coding number $a_i$ suitable for $\alpha$, enumerate $a_i$ into $A_i$, and end the current stage.

Case 4.2: $a_i$ is defined and $a_i \notin \Psi^{\Omega_i \cap A_j}$: Let $\alpha \prec \langle \text{wait} \rangle$ be eligible to act next and set the stream $S(\alpha \prec \langle \text{wait} \rangle) = [s_0, s)$, where $s_0$ is the stage at which $a_i$ was chosen.

Case 4.3: $a_i$ is defined and $a_i \in A_j \cap \Psi^{\Omega_i \cap A_j}$: Then $\alpha$ stops the strategy by extracting $a_i$ from $A_i$, and ending the current stage.

Case 4.4: $a_i$ is defined and $a_i \in \Psi^{\Omega_i \cap A_j} - A_j$: Let $\alpha \prec \langle \text{stop} \rangle$ be eligible to act next and set the stream $S(\alpha \prec \langle \text{stop} \rangle) = [s_0, s)$, where $s_0$ is the stage at which $\alpha$ stopped.

Case 5: $R$ is an $\mathcal{F}_{1, i, j}$-requirement: Pick the first subcase which applies.

Case 5.1: $\alpha$ has not been eligible to act since its most recent initialization or some coding number $a_k$, for $k \in L$, or $\tilde{a}_k$, for $k \in R$, is not defined: For each $k \in L$ with $a_k$ undefined, and $\tilde{k} \in R$ with $\tilde{a}_k$ undefined, choose new distinct coding numbers $a_k$ and $\tilde{a}_k$ that are suitable for $\alpha$ and end the current stage.

Case 5.2: All coding numbers $a_k$, for $k \in L$ and $\tilde{a}_k$ for $k \in R$ are defined but the coding number $e_j$ is not defined: Choose $e_j$ larger than any number seen so far in the construction. For all $k \in L$ and $\tilde{k} \in R$, enumerate $a_k$ into $A_k$, $e_j$ into $E_j$, and the axioms $\langle e_j, \{a_k\} \oplus \{\tilde{a}_k\} \rangle$ into $\Lambda_{j,k,k}$. End the current stage.

Case 5.3: The coding number $e_j$ is defined and $e_j \notin \Upsilon^{\Lambda_i \oplus \hat{A}_i}$: Let $\alpha \prec \langle \text{wait} \rangle$ be eligible to act next and set the stream $S(\alpha \prec \langle \text{wait} \rangle) = [s_0, s)$ where $s_0$ is the stage at which $e_j$ was chosen.

Case 5.4: The coding number $e_j$ is defined and $e_j \in E_j \cap \Upsilon^{\Lambda_i \oplus \hat{A}_i}$: Then $\alpha$ stops the strategy by extracting $e_j$ from $E_j$, every $a_k$ from $A_k$ for $k \neq i$, every $\tilde{a}_k$ from $\hat{A}_k$ for $\tilde{k} \neq i$, and ending the current stage.

Case 5.5: The coding number $e_j$ is defined and $e_j \in \Upsilon^{\Lambda_i \oplus \hat{A}_i} - E_j$: Let $\alpha \prec \langle \text{stop} \rangle$ be eligible to act next and set the stream $S(\alpha \prec \langle \text{stop} \rangle) = [s_0, s)$, where $s_0$ is the stage at which $\alpha$ stopped.

Case 6: $R$ is a $\mathcal{T}_{\Psi,j}$-requirement: Let $\beta_0 \subset \beta_1 \subset \cdots \subset \beta_{i_0-1} \subset \alpha$ be all the strategies such that some $S_i$ is active along $\alpha$ via $\beta_i$ (allowing $i_0 = 0$). For every $i < i_0$, and for every $x$ that has been dumped into $A_j$, enumerate the axiom $\langle x, \emptyset \rangle$ into $\Delta_i$. (In the following subcases, the enumeration operators $\Omega_i$ and $\Gamma_i$ are those of $\beta_i$, for $i \in [0, i_0]$.)

Pick the first case which applies.

Case 6.1: $\alpha$ has not been eligible to act since its most recent initialization: For each $i \in [0, i_0)$, pick killing points $q_i$ larger that any number seen so far in the construction and
enumerate \( q_i \) into \( B \). End the current stage \( s \).

**Case 6.2.** \( \alpha \) has current killing points but the witness \( z_{i_0} \) is undefined: Choose \( z_{i_0} \) suitable for \( \alpha \), add \( z_{i_0} \) to \( A_j \), initialize all strategies \( \beta \supseteq \alpha^- (\text{wait}) \), and end the current stage.

**Case 6.3.** \( z_{i_0} \) is defined and \( z_{i_0} \notin \Phi^B \): End the substage by letting \( \alpha^- (\text{wait}) \) be eligible to act next and setting the stream \( S(\alpha^- (\text{wait})) = S(\alpha) \cap [s_0, s) \) where \( s_0 \) is the stage at which \( z_{i_0} \) was chosen.

**Case 6.4.** \( \alpha \) has stopped (as defined below) and has not been initialized since then: End the substage by letting \( \alpha^- (\text{stop}) \) be eligible to act next and setting the stream \( S(\alpha^- (\text{stop})) = S(\alpha) \cap [s_0, s) \), where \( s_0 \) is the stage at \( \alpha \) stopped.

**Case 6.5.** Otherwise \( z_{i_0} \in A_j \cap \Phi^B \): We call \( z_{i_0} \) a realized witness. We now distinguish two subcases.

**Case 6.5.1** \( i_0 = 0 \): Then \( \alpha \) stops by extracting \( z_{i_0} \) from \( A_j \) and ending the current stage \( s \).

**Case 6.5.2** Otherwise \( i_0 > 0 \): Then \( \alpha \) stops as follows:

**Definition 6.2.** For \( i \in [0, i_0) \), call \( z \) \( \Gamma_i \)-cleared if

\[
\Gamma_i^B [s_z] \subseteq \Omega_i^{A-(z,j)} ,
\]

where \( s_z \) is the stage at which \( z \) became a realized witness of \( \alpha \).

\( \alpha \) first extracts \( z_{i_0} \) from \( A_j \). We then have further subcases depending on whether we have a witness which is “fully \( \Gamma \)-cleared.”

**Case 6.5.2.1** Some witness \( z \in S(\alpha) \) (current or former uncancelled, picked since \( \alpha \)’s most recent initialization) is \( \Gamma_i \)-cleared for all \( i \in [0, i_0) \): Then \( \alpha \) stops by removing \( z \) from \( A_j \) (if necessary), adding \( B[s_z] \) into \( B \), setting \( z_{i_0} = z \) as its current witness and ending the current stage \( s \).

**Case 6.5.2.2** Otherwise: We will now define the streams associated with \( \alpha \)’s infinitary outcomes. We will use \( z_i \) to denote the least element of the stream \( S(\alpha^- (\infty_i)) \).

\( \alpha \) acts as follows: Fix the least \( i < i_0 \) for which there is a current or former uncancelled witness \( z \) (minimal for this \( i \), picked since \( \alpha \)’s most recent initialization) such that:

\[
z \notin S(\alpha^- (\infty_i)) \]

\( z \) is \( \Gamma_k \)-cleared for all \( k \in (i, i_0) \), and

\[
z > \max \{ z_k : k \leq i \text{ and } z_k \text{ currently defined} \} .
\]

(Here we set \( \max(\emptyset) = -1 \). Note that the above condition holds trivially for \( z = z_{i_0} \) and \( i = i_0 - 1 \), so \( z \) as defined above must exist.)

Then \( \alpha \)

1. extracts \( q_k \) (for each \( k \in [i, i_0) \)) from \( B \);
2. picks new \( q_k \) (for each \( k \in [i, i_0) \)) larger than any number seen so far in the construction and enumerates them into \( B \);
3. cancels \( \Delta_k \) for all \( k \in (i, i_0) \);
4. cancels all (former or current) witnesses \( z' \neq z \) of \( \alpha \) with \( z' \notin S(\alpha^- (\infty_k)) \), for all \( k \in (i, i_0) \) makes \( z_k \) undefined, and sets \( S(\alpha^- (\infty_k)) = \emptyset \);
5. adds \( z \) to \( S(\alpha^- (\infty_j)) \) and sets \( z_i = z \) if \( z_i \) is currently undefined;
6. adds the axiom \( \langle z, \Gamma_i^B [s_z] \rangle \) into \( \Delta_i \);
7. adds axioms \( \langle z', \emptyset \rangle \) into \( \Delta_i \) for all \( z_i < z' < \max (S(\alpha^- (\infty_i))) \) with \( z' \in A_j - S(\alpha^- (\infty_i)) \); and
8. ends the substage by letting \( \alpha^- (\infty_i) \) be eligible to act next.
Ending the stage $s$: If the stage $s$ ended at Case 3.4, 4.3, 5.4, 6.5.1 or 6.5.2.1, let $z_i$ be the number extracted by the strategy $f_s$ from $A_i$. For every $\beta \subseteq f_s$, if $\beta$ is a $T_{\Phi,i}$-requirement then for every $x \in S(\beta)$, if $x > z_i$, dump $x$ into $A_i$.

For every $\alpha > L_{f_s}$, if $\alpha$ is an $E_{\jmath,i}$-strategy, and $\alpha$’s diagonalization witness $e_j$ is defined, enumerate $e_j$ into $E_j$ and the axiom $\langle e_j, \emptyset \rangle$ into $\Lambda_{\jmath,k}$, for all $j, k \in L$. Initialize every strategy $\alpha > L_{f_s}$.

The verification

Let $f = \liminf_s f_s$ be the true path of the construction, defined more precisely by induction by

$$f(n) = \liminf \{s: f_s(n) \}. $$

Lemma 7.1. (Tree Lemma)

(i) Each $\alpha \subset f$ is initialized at most finitely often.

(ii) For each strategy $\alpha \subset f$, the stream $S(\alpha)$ is an infinite set. No number can leave $S(\alpha)$ unless $\alpha$ is initialized. For every $X \in \{A, A_0, \ldots, A_n\} \cup \{\tilde{A}, \tilde{A}_0, \ldots, \tilde{A}_n\}$ and every stage $s$, there are an $\alpha$-stage $t > s$ and a number $z > s$ such that $z$ is suitable for $\alpha$ to enumerate into $X$ at stage $t$.

(iii) The true path $f$ is an infinite path through $T$.

(iv) For any requirement $R_e = S_\Omega$ or $T_{\Phi,j}$, there is a strategy $\alpha \subset f$ such that the requirement is active via $\alpha$ along all sufficiently long $\beta \subset f$, or is satisfied via $\alpha$ along all $\beta$ with $\alpha \subset \beta \subset f$. (In particular, for any requirement $R_e$, there is a longest strategy assigned to $R_e$ along $f$.)

Proof. (i) Proceed by induction on $\alpha$ and note that the only time a strategy is initialized is when it is to the left of the true path or in Case 6.2, which can only happen finitely often.

(ii) Proceed by induction on $|\alpha|$ and note for the last part of (ii) that any number just entering $S(\alpha)$ is suitable for $\alpha$ at that stage.

(iii) A stage $s$ is ended before substage $s$ only under Cases 3.1, 3.2, 3.4, 4.1, 4.3, 5.1, 5.2, 5.4, 6.1, 6.2, or 6.5.1. By (ii), we cannot stop cofinitely often at 3.1, 4.1, 5.1 or 6.2 due to lack of suitable numbers.

(iv) By an easy induction argument on $e$.

We now verify the satisfaction of the requirements.

Lemma 7.2. ($J$-Lemma) The $J$-requirement is satisfied.

Proof. Immediate from the definition of $A$.

Lemma 7.3. ($I$-Lemma) All $I$-requirements are satisfied.

Proof. Fix a requirement $I_{\Psi,j}$. By the Tree Lemma (Lemma 7.1(iv)), there is an $I$-strategy $\alpha \subset f$ such that $I_{\Psi,j}$ is satisfied along all $\beta$ with $\alpha \subset \beta \subset f$. Then $\alpha \wedge \langle o \rangle \subset f$ for $o \in \{\text{stop, wait}\}$.

By the construction, the fact that $\alpha$ is eventually no longer initialized, and the Tree Lemma (Lemma 7.1(ii)), $\alpha$ eventually has a fixed diagonalization witness. Call this witness $z$.
If \( \alpha^\prec \langle \text{wait} \rangle \subseteq f \) then \( z \in A_i - \Psi^{\bigoplus_{j \neq i} A_j} \) by the construction, thus the requirement \( \mathcal{I}_{\Psi, i} \) is clearly satisfied.

Otherwise \( \alpha^\prec \langle \text{stop} \rangle \subseteq f \), so \( \alpha \) stops at some stage \( s \), and \( z \in \Psi^{\bigoplus_{j \neq i} A_j}[s] - A_i \). We will show that no set changes at any number \( < s_z \) (where \( s_z \) is the stage \( \leq s \) at which \( z \) became a realized witness) by considering all possible strategies \( \beta \).

**Case A:** \( \beta <_L \alpha \): Then \( \beta \) is no longer eligible to act after stage \( s \) (or else \( \alpha \) would be initialized and lose its witness).

**Case B:** \( \beta > \alpha^\prec \langle \text{stop} \rangle \): The first time \( \beta \) is eligible to act after \( \alpha \) stops is the first time \( \beta \) is eligible to act after being initialized: Thus \( \beta \) cannot change \( \bigoplus_{j \neq i} A_j \) at any number that would injure \( \Psi^{\bigoplus_{j \neq i} A_j} \).

**Case C:** \( \beta^\prec \langle o \rangle \subseteq \alpha^\prec \langle \text{stop} \rangle \) for some \( o \in \{ \text{stop, wait} \} \): Then \( \beta \) cannot change \( \bigoplus_{j \neq i} A_j \) without initializing \( \alpha \).

**Case D:** \( \beta^\prec \langle \infty_i \rangle \subseteq \alpha \) for some \( i \in \omega \): Then \( z \) was put by \( \beta \) into the stream of \( \beta^\prec \langle \infty_i \rangle \), and at stage \( s \), \( \beta \) adds a number \( > z \) into the stream of \( \beta^\prec \langle \infty_i \rangle \). At the first \( \beta \)-stage \( s' > s \), \( \beta \) picks a coding number \( z' \) which is too large to injure \( \Psi^{\bigoplus_{j \neq i} A_j}(z) \), and after stage \( s \), \( \beta \) does not change \( \bigoplus_{j \neq i} A_j \) at a number less than \( z' \). So \( \beta \) cannot injure \( \Psi^{\bigoplus_{j \neq i} A_j}(z) \) after stage \( s \).

**Case E:** \( \beta^\prec \langle \text{so} \rangle \subseteq \alpha \) is an \( S \)-requirement: Then \( \beta \) never extracts any elements from \( \bigoplus_{j \neq i} A_j \).

**Lemma 7.4. (L-Lemma)** All \( L \)-requirements are satisfied.

**Proof.** The proof that \( L_{\Pi, x} \)-requirements are satisfied is similar to the proof of Lemma 7.3.

**Lemma 7.5. (N- and P-Lemma)** All \( N \)- and \( P \)-requirements are satisfied.

**Proof.** Fix a requirement \( N_{\Xi, i} \). The proof that \( N_{\Xi, i} \) is satisfied is similar to the proof of Lemma 7.3 with the additional case that when, if ever, \( N_{\Xi, i} \) extracts \( c \) from \( C \), the \( P \)-requirements will extract \( a_j \) from \( A_j \) for all \( j \in L - \{ i \} \). However, it is immediate that this action does not injure the \( \Xi_{A_i}(c) \) computation.

Fix a requirement \( P_{i, j} \) and fix some element \( c \) that was targeted to enter \( C \) by some \( N_{\Xi, k} \)-strategy \( \alpha \) at, say, stage \( s \). If \( \alpha \) was ever initialized at some stage \( s_0 > s \), then by the action at the end of stage \( s_0 \), \( c \) is enumerated into \( C \) and the axiom \( \langle c, \emptyset \rangle \) into \( \Theta_{i, j} \). In addition, \( c \) will never be chosen again as a diagonalization number by any other \( N \)-strategy.

Assume that \( \alpha \) was never initialized after stage \( s \). We have two cases to consider.

**Case 1:** \( c \in C \): At some stage \( s_1 \geq s \) we enumerated the axiom \( \langle c, \{ a_i \} \oplus \{ a_j \} \rangle \) into \( \Theta_{i, j} \), the elements \( a_i \) into \( A_i \), \( a_j \) into \( A_j \), and \( c \) into \( C \). Since \( \alpha \) was not initialized after stage \( s_1 \), no other strategy could extract either \( a_i \) from \( A_i \) or \( a_j \) from \( A_j \) without initializing \( \alpha \), and hence \( c \notin \Theta_{i, j}^{A_i \oplus A_j} \).

**Case 2:** \( c \notin C \): We have two subcases to consider.

**Case 2a:** \( c \) was never enumerated into \( C \) by \( \alpha \): By Lemma 7.1(ii), we must have \( \alpha <_L f_{s_1} \) for all stages \( s_1 > s \), and hence no axiom of the form \( \langle c, \{ a_i \} \oplus \{ a_j \} \rangle \) was enumerated into \( \Theta_{i, j} \). Therefore \( c \notin \Theta_{i, j}^{A_i \oplus A_j} \).

**Case 2b:** Otherwise: This case is similar to Case 1. At some stage \( s_1 \geq s \) we enumerate the axiom \( \langle c, \{ a_i \} \oplus \{ a_j \} \rangle \) into \( \Theta_{i, j} \), the elements \( a_i \) into \( A_i \), \( a_j \) into \( A_j \) and \( c \) into \( C \). Then, at some later stage \( s_2 > s_1 \), \( c \) is extracted from \( C \) by \( \alpha \), and so \( a_i \) is extracted from \( A_i \) or \( a_j \) from \( A_j \). Since \( \alpha \) was not initialized after stage \( s_2 \), and by Lemma 7.1(ii), no other strategy could enumerate either \( a_i \) back into \( A_i \) or \( a_j \) back into \( A_j \), and hence \( c \notin \Theta_{i, j}^{A_i \oplus A_j} \).

**Lemma 7.6. (E- and F-Lemma)** All \( E \)- and \( F \)-requirements are satisfied.
Proof. The proof that all $E_{i,t}^j$- and $F_{i,t,s}^j$-requirements are satisfied is similar to the proof of Lemma 7.5.

Lemma 7.7. (T-Lemma) All $T$-requirements are satisfied.

Proof. Fix a requirement $T_{\Phi,j}$. The proof that $T_{\Phi,j}$ is satisfied is similar to the proof of Lemma 7.3. The difference is in how we handle Case $E$.

Case $E$: $\beta \in (\alpha \cup s)$ and $\beta$'s $S$-requirement is active along $\alpha$ via $\beta$: Then $\alpha$ stops via Case 6.5.2.1 of the construction where $\beta = \beta_i$ for some $\beta_i$ mentioned in Case 6.5.2.2. Thus $z$ is $\Gamma_\gamma$-canceled, i.e.

$$\Gamma_i^B[s_z] \subseteq \Omega_1^{A^{-i}(z,j)},$$

where $s_z$ is the stage at which $z$ became a realized witness of $\alpha$. By the action at stage $s$ (the stage when $T_{\Phi,j}$ stops),

$$\Gamma_i^B[s] \subseteq \Omega_1^A[s],$$

so any later $\Gamma_\gamma$-correction by $\beta$ will only involve $\Gamma_\gamma$-axioms defined after stage $s_z$, and thus will change any set only on numbers $> s_z$.

To complete this lemma, we add an additional case.

Case $F$: $\beta \in (\alpha \cup s)$ and $\beta$'s $S$-requirement is not active along $\alpha$ via $\beta$: Then some $\alpha'$ with $\beta \subseteq \alpha' \subseteq \alpha$ kills $\beta$'s enumeration operator $\Gamma$. Therefore

$$\Gamma_i^B[s_z] \subseteq \Omega_1^A[s_z]$$

by the action of $\beta$ at stage $s_z$. Any later $\Gamma$-correction performed by $\beta$ will only involve $\Gamma$-axioms defined after stage $s_z$, and hence will change any set only on numbers $> s_z$.

Lemma 7.8. (S-Lemma) All $S$-requirements are satisfied.

Proof. Fix a requirement $S_\Omega$. By the Tree Lemma (Lemma 7.1(iv)), there is a longest $S_\Omega$-strategy $\beta \in f$. Again by the Tree Lemma (Lemma 7.1(iv)), we may now distinguish two cases:

Case 1: $S_\Omega$ is active via $\beta$ along all $\alpha$ with $\beta \subseteq \alpha \subseteq f$: Suppose that $\beta$ is no longer initialized after, say, stage $s_0$.

For the sake of a contradiction, assume first that there is some $z \in \Omega_1 - \Gamma_i^B$. Choose $s_0$ to be the least oldest such $z$ with age $s_z$. Fix $s_1 \geq s_0, s_z$ such that no $T$-strategy with killing point $\leq z$ (for this $\Gamma_i$) executes Step (i) of Case 6.5.2.2 of the construction. Then by the first $\beta$-expansionary stage $\geq s_1$, $\beta$ will permanently put $z$ into $\Gamma_i^B$ by Case 1 of the construction.

If $z \in \Gamma_i^B$, then by $\Gamma$-correction of $\beta$ under Case 1 of the construction, $z \in \Omega_1$.

Case 2: There is a $T_{\Phi,j}$-strategy $\alpha \in f$ such that $S_\Omega$ is satisfied via $\alpha$ along all $\xi$ with $\beta \subseteq \xi \subseteq f$: Then $\beta$ is $\alpha$'s strategy $\beta_i$, $\beta_i^{-1}(\xi_i) \subseteq f_i$, and we need to show that $\Delta_i^A = A_j$ (for the enumeration operator $\Delta_i$ built by $\alpha$ after $\alpha$'s last initialization and after $\alpha$ cancels $\Delta_i$ for the last time).

We show that $A_j = * \Delta_i^{\Omega_1^A}$ by distinguishing two cases for arguments $z \geq z_i$ of $\Delta_i^{\Omega_1^A}$:

Case 2a: $z \notin S(\alpha^{-1}(\xi_i))$: Then, once $z < \max(S(\alpha^{-1}(\xi_i))[s])$, no strategy can remove $z$ from $A_j$ (and so by (7) of Case 6.5.2.2 of the construction, $z \in A_j$ if and only if $z \in \Delta_i^{\Omega_1^A}$). To see this, note that only strategies $\xi \subseteq \alpha$ with infinitary outcome along $\alpha$ can possibly change $A_j(z)$ (by the usual initialization argument). But, after stage $s$, any such $\xi$ cannot put $z$ into the stream of any strategy $\xi' \supset \xi$. If $\xi$ is a $T$- or $I$-strategy, it will no longer remove $z$ as a realized witness, and it will not remove $z$ for $\Gamma$-correction (as in Case 4.3 or Case 6.5.1 of the construction) since $\xi$ does not stop (as Case 4.3 or Case 6.5.1 does not apply). If $\xi$ is an $S$-strategy, then $\xi$ does not remove numbers from $A_j$.

17
Case 2b: $x \in S(\alpha \setminus (\infty_i))$: We first observe that

\[
\begin{align*}
    z \in A_j & \iff \langle z, j \rangle \in A, \\
    z \in \Delta_{\Omega^4}^i & \iff \Gamma_i^B \subseteq \Omega^A, \text{ and} \\
    \Gamma_i^B[s_z] & \not\subseteq \Omega^{\Delta^1_i \setminus \{\langle z, j \rangle\}}
\end{align*}
\]

by meeting the $J$-requirement, the definition of $\Delta_i$, and the fact that $\alpha$ does not stop, respectively.

Thus, if $z \not\in A_j$, by (1) and (3) we have $\Gamma_i^B[s_z] \not\subseteq \Omega^A$, which by (2), gives us $z \not\in \Delta_{\Omega^4}^i$.

On the other hand, if $z \in A_j$, then by (1), $\langle z, j \rangle \in A$ so it follows that

\[
\Gamma_i^B[s_z] \subseteq \Omega^{\Delta^1_i \setminus \{\langle z, j \rangle\}}[s_z] \subseteq \Omega^A
\]

and we have $z \in \Delta_{\Omega^4}^i$. 

\[\square\]

**Lemma 7.9.** The sets $B, \tilde{B}, C, \tilde{C}, E_0$, and $E_1$ are $\Delta^0_2$.

**Proof.** We prove that $B$ is $\Delta^0_2$. The proof for $\tilde{B}$ is similar. In the construction, only under Case 1 and Case 6.5.2.2 do we enumerate elements into or extract elements from $B$.

Fix an element $z$ and an enumeration operator $\Pi$. By Lemma 7.4, the limit $\lim_s \Pi^A(z)[s]$ converges. Hence, any $S_\Pi$-strategy that chooses a coding number $c_z$ for $z$ under Case 1 will enumerate $c_z$ into and extract $c_z$ from $B$ a finite number of times. Furthermore, we choose our coding numbers $c_z$ in such a way that if ever $S_\Pi$ is reset, no other strategy will enumerate $c_z$ into $B$.

Under Case 6.5.2.2, a killing point can be enumerated into and extracted from $B$ at most once. Like in the previous case, we choose new killing points in such a way that no killing point, once canceled, will ever be used again by another strategy. Therefore $B$ is $\Delta^0_2$.

By Lemma 7.4, $A \oplus \tilde{A}$ is low and by Lemma 7.5, $C, \tilde{C} \leq_e \tilde{A} \oplus A$. Therefore both $C$ and $\tilde{C}$ are $\Delta^0_2$. An element $x$ may be enumerated into $E_j$ at most once by Case 5.2 of the construction and extracted at most once by Case 5.4. The only other time that $x$ may be enumerated into $E_j$ is when it is dumped in due to initialization. This may happen at most once and after this, $x$ will never be extracted from $E_j$. Therefore both $E_0$ and $E_1$ are $\Delta^0_2$.

\[\square\]

This completes the proof of the theorem.

**References**


[Coo84] S. Cooper. Partial degrees and the density problem, part II: The enumeration

[Coo87] S. Cooper. Enumeration reducibility using bounded information: counting minimal
covers. *Zeitschrift für mathematische Logik und die Grundlagen der Mathematik*,


[LN95] S. Lempp and A. Nies. The undecidability of the $\Pi_4$-theory for the r.e. wtt- and

Turing degrees is undecidable. *Transactions of the American Mathematical Society*,


[LS02] S. Lempp and A. Sorbi. Embedding finite lattices into the $\Sigma^0_2$ enumeration degrees.

[LSS] S. Lempp, T. Slaman, and S. Sorbi. Extensions of embeddings in the $\Sigma^0_2$ enumeration
degrees. Accepted for publication in the Journal of Mathematical Logic.


