

SLENDER CLASSES.

ROD DOWNEY AND ANTONIO MONTALBÁN

Definition 0.1. A Π_1^0 class S is *slender* if for every other Π_1^0 class F , there exists a clopen set C such that

$$\text{int}(F) \cap \text{iso}(S) = C \cap \text{iso}(S)$$

0.1. **Notation.** If S is a tree, $S_\tau = \{\sigma \in S, \sigma \subseteq \tau \vee \tau \subseteq \sigma\}$.

$$S[s] = S \cap 2^{\leq s}$$

$[S][s]$ = $S \cap 2^s$, the stage- s approximation to $[S]$.

$\tau \in S$ is *dead at stage s* if it has no extensions in $[S][s]$.

$[\tau] = \{X \in 2^\omega : \tau \subset X\}$ and also $[\tau] = (2^{<\omega})_\tau$.

\emptyset is the empty string.

1. PERFECT AND THIN VERSUS FINITE

Theorem 1.1. *There is a uniform procedure which, given a computable tree $T \subseteq 2^{<\omega}$, builds a computable tree S such that, if $[T]$ is perfect $[S]$ is perfect and thin, and if $[T]$ is not perfect $[S]$ is finite.*

This theorem will be used in both constructions, the one of a non-slender thin class, and the one of a slender thin class. The proof starts developing ideas that will be used in both of those constructions.

In the case when $[T]$ is perfect, we will define two tree-embeddings $f, r: 2^{<\omega} \rightarrow S$ satisfying that for every $\sigma \in 2^{<\omega}$,

(fr1) $f(\sigma) \subseteq r(\sigma)$, $f(\sigma \hat{\ } 0) = r(\sigma) \hat{\ } 0$, $f(\sigma \hat{\ } 1) = r(\sigma) \hat{\ } 1$, and

(fr2) $[S_{f(\sigma)}] = [S_{r(\sigma)}]$

It follows that $[S] = [\text{image}(f)] = [\text{image}(r)]$, and hence that $[S]$ is perfect.

There are two types of requirements: the thinness requirements

$$\mathcal{T}_e : F_e \subseteq S \Rightarrow \exists C \subseteq 2^\omega \text{ clopen } ([F_e] = [S] \cap C),$$

and the finiteness requirements

$$\mathcal{F}_e : [T_{t_e}] \text{ is isolated } \Rightarrow [S] \text{ is finite.}$$

where $\{t_0, t_1, \dots\}$ is an enumeration of T . These requirements are subdivided even further. Each requirement \mathcal{T}_e is divided into 2^{2^e} sub-requirements \mathcal{T}_σ , one for each $\sigma \in 2^{2^e}$.

$$\mathcal{T}_\sigma : \text{ either } [(F_e)_{f(\sigma)}] = [S_{f(\sigma)}], \text{ or } [F_e] \cap [S_{f(\sigma)}] = \emptyset.$$

Note that if all the requirements \mathcal{T}_σ for $\sigma \in 2^{2^e}$ are satisfied, then so is \mathcal{T}_e by letting $C = \bigcup \{[f(\sigma)] : \sigma \in 2^{2^e} \ \& \ [(F_e)_{f(\sigma)}] = [S_{f(\sigma)}]\}$. The strategy of \mathcal{T}_σ is roughly the following. If \mathcal{T}_σ sees the opportunity to define $r(\sigma) \notin F_e$, it will do it guaranteeing that $F_e \cap S_{r(\sigma)} = \emptyset$. If such an opportunity never appears, it is because $(F_e)_{f(\sigma)} = S_{f(\sigma)}$.

The author supported by NSF Grant DMS-0600824 and by the Marsden Fund of New Zealand, via postdoctoral fellowships.

Each requirement \mathcal{F}_e is to be divided into 2^{2e+1} sub-requirements \mathcal{F}_σ , one for each $\sigma \in 2^{2e+1}$.

\mathcal{F}_σ : if $[T_{t_e}]$ is isolated $\Rightarrow [S_{f(\sigma)}]$ is isolated.

\mathcal{F}_σ works roughly as follows. Every time it believes $[T_{t_e}]$ is isolated, it will kill all the paths in $[S_{f(\sigma)}][s]$, except for one. If this occurs infinitely often it is because $[T_{t_e}]$ is isolated and it will make $[S_{f(\sigma)}]$ isolated too. Otherwise, after some stage it won't act anymore and let the construction above $f(\sigma)$ continue.

1.1. Organization of the construction. We will define a computable tree S by stages; at stage s we will define $S[s] = S \cap 2^{\leq s}$. The functions f and r are also defined by stages and their values might change along the construction. At the end of stage s , we will have $f[s]$ and $r[s]$ defined on a finite tree $D_s \subset 2^{<\omega}$. We will always have that, if $\{d_0, \dots, d_k\}$ is the set of end-nodes of D_s , then $[S][s] = \{r(d_0)[s], \dots, r(d_k)[s]\}$.

Each $\sigma \in 2^{<\omega}$ has a requirement \mathcal{R}_σ (either \mathcal{T}_σ or \mathcal{F}_σ) assigned. If $\sigma, \tau \in 2^{<\omega}$ are incomparable, then the requirements \mathcal{R}_σ and \mathcal{R}_τ do not interact at all with each other, and none of the two requirements has stronger priority than the other one. If $\sigma \subset \tau$, then \mathcal{R}_σ has stronger priority than \mathcal{R}_τ and it is allowed to cancel it. Cancellation of \mathcal{R}_τ by \mathcal{R}_σ is all the interaction there is between \mathcal{R}_σ and \mathcal{R}_τ . Requirement \mathcal{R}_σ is responsible for defining $f(\sigma)$ and $r(\sigma)$ satisfying conditions (fr1) and (fr2).

At each stage $s+1$, we will start by *activating* the strategy for \mathcal{R}_\emptyset . This strategy might later activate $\mathcal{R}_{(0)}$ and then $\mathcal{R}_{(1)}$. Then, $\mathcal{R}_{(0)}$ could activate $\mathcal{R}_{(0,0)}$ and $\mathcal{R}_{(0,1)}$ and so on. In general, when a requirement \mathcal{R}_σ is activated, either it *acts* and then ends by canceling all the requirements of lower priority and stop going up the tree for now, or it does not act and activates $\mathcal{R}_{\sigma \smallfrown 0}$ and $\mathcal{R}_{\sigma \smallfrown 1}$. In the former case we have that σ is an end node of D_s , so \mathcal{R}_σ is also responsible for defining $[S_{f(\sigma)}][s+1]$ so that $[S_{f(\sigma)}][s+1] = \{r(\sigma)[s+1]\}$. Since there is no interaction between $\mathcal{R}_{\sigma \smallfrown 0}$ and $\mathcal{R}_{\sigma \smallfrown 1}$, it does not matter whether they run simultaneously or one after the other one. At each stage we traverse a finite part of $2^{<\omega}$, namely D_s .

In the case when $[T]$ is perfect, the construction will be a finite injury one. Every requirement will be activated infinitely often, but it will stop canceling weaker priority ones after some stage and $f[s]$ and $r[s]$ will reach a limit. When $[T]$ is not perfect, and t_e is the first node such that $[T_{t_e}]$ is isolated, every requirement \mathcal{R}_σ with $|\sigma| > 2e+1$ will be canceled infinitely often. However, in this case, requirement \mathcal{F}_e is the only one that needs to be satisfied.

The first time \mathcal{R}_σ is active (either first time ever or first time since it was last cancelled), it has to be *initialized*. \mathcal{R}_σ defines $f(\sigma)$ using (fr1): If $\sigma = \tau \smallfrown i$, then $f(\sigma)[s+1] = r(\tau)[s+1] \smallfrown i$. (If $\sigma = \emptyset$, let $f(\sigma) = \emptyset$.) At the beginning, it defines $r(\sigma)[s+1] = f(\sigma)[s+1]$. It also sets its status to an initial status that depends on the requirement. We observe that necessarily $f(\sigma)[s+1] = r(\sigma)[s+1]$ have length $s+1$: Since \mathcal{R}_σ became activated at this stage, it means that for every $\gamma \subset \sigma$, \mathcal{R}_γ did not act, and hence $r(\gamma)[s] = r(\gamma)[s+1]$. Another observation is that since \mathcal{R}_σ was not active at stage s , it means that $\sigma \notin D_s$, but $\tau = \sigma^-$ had been initialized before s , so $\tau \in D_s$. Hence $r(\tau)[s] = r(\tau)[s+1]$ has length s , and both $f(\sigma)[s+1] = r(\sigma)[s+1]$ have length $s+1$. So, we cannot continue going up $2^{<\omega}$ until the next stage. The value of $f(\sigma)$ will not change again, unless \mathcal{R}_σ is canceled by a stronger priority requirement. The value of $r(\sigma)$ might change a few times before

stabilizing. Every time $r(\sigma)$ changes, \mathcal{R}_σ initialices all the weaker priority requirements, that is, all the \mathcal{R}_τ with $\tau \supset \sigma$.

We now describe the strategies of the requirements \mathcal{R}_σ .

1.2. Thinness requirement. Consider $\sigma \in 2^{<\omega}$, $|\sigma| = 2e$. Recall that \mathcal{T}_σ is the requirement: either $[(F_e)_{f(\sigma)}] = [S_{f(\sigma)}]$, or $[F_e] \cap [S_{f(\sigma)}] = \emptyset$. Suppose we are at stage $s+1$ and \mathcal{T}_σ has been activated. Also assume that \mathcal{T}_σ has been initialized in some previous stage. So $f(\sigma)$ and $r(\sigma)$ have been previously defined, and \mathcal{T}_σ is in status either **wai** (for “waiting”) or **sat** (for satisfied). The initial status is **wai**.

First let us assume the current status of \mathcal{T}_σ is **wai**. Check whether $[(F_e)_{f(\sigma)}][s] = [S_{f(\sigma)}][s]$.

- If so, we keep the status **wai** and pass control to requirements $\mathcal{F}_{\sigma \smallfrown 0}$ and $\mathcal{F}_{\sigma \smallfrown 1}$. We will check again at the next stage \mathcal{T}_σ gets activated. If we have to keep on waiting for ever, the requirement is satisfied because $(F_e)_{f(\sigma)} = S_{f(\sigma)}$.
- Otherwise we *act*. Consider $\gamma \in [S_{f(\sigma)}] \setminus [F_e][s]$. We let $r(\sigma) = \gamma \smallfrown 0$ and $[S_{f(\sigma)}][s+1] = \{r(\sigma)\}$, making sure that $[F_e] \cap [S_{f(\sigma)}] = [F_e] \cap [S_{r(\sigma)}] = \emptyset$. The status of \mathcal{T}_σ is set to **sat**. All the requirements \mathcal{R}_τ for $\tau \supset \sigma$ are canceled. Unless \mathcal{T}_σ is canceled at some later stage, it is satisfied for ever and this is the only one time \mathcal{T}_σ cancels lower priority requirements. We do not need to continue going up $2^{<\omega}$ at this stage.

If \mathcal{T}_σ is in status **sat** when it is activated, it immediately passes control to $\mathcal{F}_{\sigma \smallfrown 0}$ and $\mathcal{F}_{\sigma \smallfrown 1}$.

1.3. Finiteness requirements. Consider $\sigma \in 2^{<\omega}$, $|\sigma| = 2e + 1$. Recall that \mathcal{F}_σ is the requirement: if $[T_{t_e}]$ is isolated, $[S_{f(\sigma)}]$ is isolated, where $t_e \in T$. We say that $n > |t_e|$ is *verified at s* if exactly one string $\tau \in T_{t_e} \cap 2^n$ is not dead at stage s . So, we have that $[T_{t_e}]$ is isolated if and only if every $n > |t_e|$, there exists a stage s at which n is verified. The strategy for \mathcal{F}_σ is to try to verify every $n > |t_e|$. Suppose that we are at stage $s+1$. At the beginning \mathcal{F}_σ sets $n_\sigma[s+1] = |t_e| + 1$, and starts waiting for a stage verifying $n_\sigma[s+1]$. At later stages, there will be some other value of $n_\sigma > |t_e|$, which is waiting to be verified.

- (1) If n_σ gets verified as s , we momentarily believe that $[T_{t_e}]$ is isolated. We define $S[s+1]$ so that only one string in $[S][s]$ is extended to $[S][s+1]$ and we let $r(\sigma)$ be that one extension. We add one to the value of n_σ , and the next stage \mathcal{F}_σ gets activated, we will be waiting for the new n_σ to be verified. We then cancel all the requirement \mathcal{R}_τ for $\tau \supset \sigma$ and we stop going up $2^{<\omega}$ for this stage. The status of \mathcal{F}_σ is set to **iso** for isolated.
- (2) If n_σ does not get verified, then we pass control to $\mathcal{T}_{\sigma \smallfrown 0}$ and $\mathcal{T}_{\sigma \smallfrown 1}$. The status of \mathcal{F}_σ is set to **niso** for not isolated.

Note that if $[T]$ is perfect, then for every t_e , there will be some n_e which will never be verified. After that n_e is chosen by \mathcal{F}_σ , \mathcal{F}_σ will never act again and let the lower priority requirement do their work. Also, if $[T_{t_e}]$ is empty, there will also be some n_e which will never be verified. On the other hand, if $[T]$ is neither perfect nor empty, for some t_e , $[T_{t_e}]$ is isolated. For each requirement \mathcal{F}_σ , $\sigma \in 2^{2e+1}$, every $n > |t_e|$ will be verified at some stage, and hence there be inifinely many stages with $[S_{f(\sigma)}][s]$ having only one element. So, $[S_{f(\sigma)}]$ will consist of an isolated path. \mathcal{F}_σ will keep on injuring the requirements \mathcal{R}_τ

for $\tau \supset \sigma$. But, since we are assuming that $S_{f(\sigma)}$ is isolated, we do not need to worry about them.

(We mentioned the status of \mathcal{F}_σ only because we will use it in the Section 3)

2. A THIN, NON-SLENDER CLASS

Theorem 2.1. *For every Δ_2^0 Boolean algebra \mathcal{B} with infinitely many atoms, there exists a thin but not slender computable tree S whose lattice of sub- Π_1^0 -classes is isomorphic to \mathcal{B} .*

Definition 2.2. Given a set $X \subseteq 2^\omega$, the *algebra of clopen set of X* , $\text{clo}(X)$ is the Boolean algebra whose elements are of the form $C \cap X$, where C is a clopen subset of 2^ω .

If T is a computable tree, we write $\text{clo}(T)$ for $\text{clo}([T])$.

Note that if $[T] \subseteq 2^\omega$ is a thin Π_1^0 class, then $L([T] \downarrow)$ coincides with the algebra of clopen sets of $[T]$.

Also observe that $\text{clo}(T)$ is isomorphic to the Boolean algebra of clopen sets of 2^ω modulo the equivalence relation $C \equiv D \Leftrightarrow C \cap [T] = D \cap [T]$, and that the elements of $\text{clo}(2^\omega)$ can be represented by finite sets of binary strings.

Lemma 2.3. *For every Δ_2^0 Boolean algebra \mathcal{B} , there is a computable tree T whose algebra of clopen sets is \mathcal{B} .*

Fix such a computable tree T . To simplify notation, assume without loss of generality that if one of $\sigma \frown 0, \sigma \frown 1$ in T , then both are.

We build a computable tree S and two tree-embeddings $f, r: T \rightarrow S$ satisfying condition **fr1** for $\sigma \in T$. Unfortunately, we will not have $[S] = [\text{image}(f)] = [\text{image}(r)]$ as in the previous construction. Instead, we will construct S, f and r with the following properties.

- (Sfr1) If $[T_\sigma]$ is empty, then so is $[S_{f(\sigma)}]$;
- (Sfr2) If $[T_\sigma]$ is isolated, then $[S_{f(\sigma)}]$ is finite;
- (Sfr3) If $[T_\sigma]$ has more than one element and is not perfect, then $[S_{f(\sigma)}] = X_\sigma \cup [S_{r(\sigma)}]$, where X_σ is a finite set disjoint from $[r(\sigma)]$.
- (Sfr4) If $[T_\sigma]$ is perfect, then $[S_{f(\sigma)}] = X_\sigma \cup [S_{r(\sigma)}]$, where X_σ is either perfect or empty, and is disjoint from $[r(\sigma)]$.

Using Remmel-Vought's theorem, we can prove that these conditions imply that $\text{clo}(S) \cong \text{clo}(T) \cong \mathcal{B}$. If we also manage to make S thin, we will have that $L(S \downarrow) \cong \mathcal{B}$.

Theorem 2.4 (Remmel-Vought [?]). *Let \mathcal{B}_0 and \mathcal{B}_1 be Boolean algebras with infinitely many atoms. Suppose $\varphi: \mathcal{B}_0 \rightarrow \mathcal{B}_1$ is a Boolean algebra embedding such that \mathcal{B}_1 is generated by the image of φ and some atoms, and that every atom of φ is mapped to a finite sum of atoms in \mathcal{B}_1 . Then, \mathcal{B}_0 and \mathcal{B}_1 are isomorphic.*

Lemma 2.5. *If S and f satisfy the conditions above, then the clopen Boolean algebra of S is isomorphic to the one of T , namely \mathcal{B} .*

PROOF: We define a map $\varphi: \text{clo}(T) \rightarrow \text{clo}(S)$ which satisfies the conditions in Vought-Remmel's theorem. If $[S_{f(\sigma)}] = X_\sigma \cup [S_{r(\sigma)}]$, where X_σ is a finite set, it is because $[S_{r(\sigma)}]$ contains at least one isolated path; choose one and call it I_σ . The idea is to put $[S_{r(\sigma)}]$

together with I_σ below the image under φ of some atom of $\text{clo}(T)$. We first define φ on T by recursion, and extend it to $\text{clo}(T)$ in the obvious way. We abuse notation, and when we write $\varphi(\sigma)$ for $\sigma \in T$, we actually mean $\varphi([\sigma] \cap [T])$. Let $\varphi(\emptyset) = \emptyset$. Now we want to define $\varphi(\tau)$ If $[T_\tau]$ is an isolated path, and τ is the root of that path, we let

$$\varphi(\tau) = f(\tau) \cup \bigcup_{\sigma \subseteq \tau: I_\sigma = [T_\tau]} X_\sigma$$

Every extension of τ is equivalent to it in $\text{clo}(T)$, so we don't need to define φ on them. If $[T_\tau]$ is not isolated, but is not perfect either, define $\varphi(\tau) = f(\tau)$. If $[T_\tau]$ is perfect, then so is $S_{f(\tau)}$. Define φ mapping $\text{clo}(T_\tau)$ to $\text{clo}(S_{f(\tau)})$ isomorphically. It is not hard to verify that φ is as wanted. \square

The construction of S , f , and r is a finite injury one, and its organization is very similar to the one in the previous section. There are three types of requirements: the thinness requirements \mathcal{T}_σ , one for every $\sigma \in T$, σ of length $3e$; the finiteness requirements \mathcal{F}_σ , one for every $\sigma \in T$, σ of length $3e + 1$; and the Non-slenderness requirements \mathcal{N}_σ , one for every $\sigma \in T$, σ of length $3e + 2$. The thinness and the finiteness requirements work exactly as the ones in the previous construction. Let us now describe how the Non-slenderness requirements work.

2.1. Non-slenderness requirements. The Non-slenderness requirement \mathcal{N} will construct a computable tree $F \supseteq S$ such that for no clopen set C we have $\text{int}([F]) \cap \text{iso}([S]) = C \cap \text{iso}([S])$. \mathcal{N} has infinitely many sub-requirements \mathcal{N}_σ , one for each $\sigma \in T$, $|\sigma|$ of the form $3e + 2$.

$$\mathcal{N}_\sigma : \quad [T_\sigma] \text{ has an isolated path} \Rightarrow \exists \text{ finite sets } Z_\sigma, Y_\sigma \subset 2^\omega \text{ such that} \\ [S_{f(\sigma)}] = Z_\sigma \cup Y_\sigma \cup [S_{r(\sigma)}] \quad \& \quad Z_\sigma \subset \text{int}([F]) \quad \& \quad Y_\sigma \cap \text{int}([F]) = \emptyset,$$

We claim that if all the requirements \mathcal{N}_σ are satisfied, then S is not slender. We know that $[T]$ has infinitely many isolated paths. Let $\{\sigma_0, \sigma_1, \dots\} \subseteq T$ an enumeration of the roots of all this isolated paths. (that is, for each σ_i , $[T_{\sigma_i}]$ is an isolated path, but $[T_{\sigma_i^-}]$ is not). For each i , let $\tau_i \subseteq \sigma_i$ be the longest string whose length is of the form $3e + 2$. Since we are assuming \mathcal{N}_{τ_i} is satisfied, we have two finite sets $Z_{\tau_i}, Y_{\tau_i} \subset \text{iso}[S] \cap [f(\tau_i)]$, such that $Z_\sigma \subset \text{int}([F])$ and $Y_\sigma \cap \text{int}([F]) = \emptyset$. Suppose toward a contradiction there is a clopen set C such that $\text{int}([F]) \cap \text{iso}([S]) = C \cap \text{iso}([S])$, and suppose that C is a finite union of basic open sets $[\pi_j]$, with $|\pi_j| < k$. Let i be such that $|\tau_i| > k$ and hence $|f(\tau_i)| > k$. Then, either $[f(\tau_i)] \subseteq C$ or $[f(\tau_i)] \cap C = \emptyset$, contradicting $Z_{\tau_i} \subseteq C$ and $Y_{\tau_i} \cap C = \emptyset$.

These requirements do not injure lower priority requirements.

At the first stage \mathcal{N}_σ is active after initialized, say $s + 1$, it defines $r(\sigma) = f(\sigma) \frown 0$. (Note that \mathcal{N}_σ had to be initialized at stage s , and hence $|f(\sigma)| = s$.) \mathcal{N}_σ also enumerates $f(\sigma) \frown 0$ and $f(\sigma) \frown 1$ into S and hence into F too. This is all it does at this stage. In the next stages it will build Z_σ and Y_σ extending $f(\sigma) \frown 10$ and $f(\sigma) \frown 11$ respectively, and will let the rest of the construction continue in top of $f(\sigma) \frown 0$. Here is how it builds Z_σ and Y_σ . Let Q_σ be the tree obtained when Theorem 1.1 is applied to T_σ . So, $[Q_\sigma]$ is perfect and thin if T_σ is perfect, and $[Q_\sigma]$ is finite otherwise. Requirement \mathcal{N}_σ places two copies of Q_σ in S and F , one in top of $f(\sigma) \frown 10$ and one in top of $f(\sigma) \frown 11$. Since it might be canceled later, \mathcal{N}_σ builds these extensions step by step. Recall that at a stage s we can only define S up to length s . \mathcal{N}_σ will include the whole cone $[f(\sigma) \frown 10]$ inside $F_{f(\sigma) \frown 10}$,

but it will let $F_{f(\sigma)\frown 11} = S_{f(\sigma)\frown 11} = Q_\sigma$. So, we will have that $Z_\sigma = [S_{f(\sigma)\frown 10}] \subseteq \text{int}([F])$ and $Y_\sigma \cap \text{int}([F]) = [S_{f(\sigma)\frown 11}] \cap \text{int}([F]) = \emptyset$. (It is not hard to see that if S is thin, then $\text{int}(S) = \emptyset$.) Also note that $X_\sigma = Z_\sigma \cup Y_\sigma$ satisfies conditions (Sfr1)-(Sfr4).

2.2. Thinness. The Thinness requirements \mathcal{T}_σ work exactly as in the previous construction and there is no problem getting them realized. However, when we showed that the satisfaction of all the \mathcal{T}_σ for $\sigma \in 2^{2e+1}$ implies the satisfaction of \mathcal{T}_e , we used the fact that $[S] = \bigcup_{\sigma \in 2^{2e+1}} [S_{f(\sigma)}]$, which is not true in this case. Now we have that

$$[S] = \bigcup_{\sigma \in T \cap 2^{2e+1}} [S_{f(\sigma)}] \cup \bigcup_{\tau \in T \cap 2^{<2e+1}} X_\tau.$$

This is not a problem since for every $\tau \in T \cap 2^{<2e+1}$, X_τ is thin.

2.3. A word on dead nodes. Some nodes $\sigma \in T$ have no extensions in $[T]$. These are the ones that we call *dead* nodes. But it might take a while to find this out, and requirement \mathcal{R}_σ will start its job as usual. After the stage s when we find out that σ is a dead node (i.e. σ has no extensions in $T[s]$), we do not need to work for \mathcal{R}_σ anymore. We could do a bit more work, if we actually want f and r to be defined at every node of T , but we do not need to worry about this later. The next time \mathcal{R}_σ becomes activated, we do not do anything, and we stop building S above $f(\sigma)$.

3. A THIN, SLENDER CLASS

Theorem 3.1. *For every Δ_2^0 Boolean algebra \mathcal{B} there exists a thin and slender computable tree S whose lattice of sub- Π_1^0 -classes is \mathcal{B} .*

Fix a computable T as the one given by Lemma 2.3. Again, to simplify notation, assume without loss of generality that if one of $\sigma \frown 0, \sigma \frown 1$ in T , both are.

This construction has three types of requirement: thinness requirements, finiteness requirements and Slenderness requirements. Each node $\sigma \in T$ will have a requirement assigned \mathcal{R}_σ that can be of any of these three kinds as in the previous constructions, and the thinness and finiteness requirements will work exactly as before. One difference is that this is an infinite injure construction, because the Slenderness requirements will have Π_2^0 and Σ_2^0 outcomes. The construction is organized on a tree of strategies, so each \mathcal{R}_σ will have a belief on the outputs of stronger priority requirements, namely $\{R_\tau : \tau \subset \sigma\}$. Actually, we will have different versions of \mathcal{R}_σ for the different possible beliefs, as one usually has in tree-of-strategies arguments, and one of these versions will act infinitely often and get injured only finitely often.

We will construct a computable tree S by stages and functions f and r satisfying (fr1), (Sfr1)-(Sfr2). We start by describing how the Slenderness requirements work, and then we will explain how the construction is organized on the tree of strategies.

3.1. Slenderness requirement. For every computable tree F_e we have a *Slenderness requirement*:

$$\mathcal{S}_e : \quad \exists \text{ clopen } C \quad (\text{int}[F_e] \cap \text{iso}[S] = C \cap \text{iso}[S]).$$

We partition these requirements into at most 2^{e+2} many requirements, one for each string $\sigma \in T$ of length $3e+2$:

$$\mathcal{S}_\sigma : \quad \text{either } \text{int}([F_e]) \cap [S_{f(\sigma)}] = \emptyset, \text{ or } [S_{r(\sigma)}] \subseteq \text{int}(F_e).$$

Note that if \mathcal{S}_σ is satisfied for every string $\sigma \in T$ of length $3e + 2$, then \mathcal{S}_e is satisfied: Let $C_0 = \bigcup\{[r(\sigma)] : \sigma \in T \cap 2^{3e+2}, [S_{r(\sigma)}] \subseteq \text{int}[F_e]\}$. Then $\text{int}[F_e] \cap \text{iso}[S] = (C_0 \cap \text{iso}[S]) \cup X$, where $X = \text{int}[F_e] \cap \bigcup_{\tau \in T \cap 2^{<3e+2}} X_\tau$. Note that (Sfr1)-(Sfr2) imply that X contains a finite number of isolated paths. Let C be the union of C_0 and those isolated paths.

We say that $\tau \in S$ is *e-verified* at stage s if $\exists \gamma \in 2^{\leq s} (\gamma \supseteq \tau \ \& \ \gamma \notin F_e)$. So, we have that $\text{int}(F_e) \cap S_{f(\sigma)} = \emptyset$ if for every $\tau \in S_{f(\sigma)}$, there is a stage s at which τ is *e-verified*. \mathcal{N}_σ will try to make sure that every $\tau \in S_{f(\sigma)}$ is *e-verified*, and if so, we say it has outcome ∞ . If instead we find a string $\tau \in S_{f(\sigma)}$ that is never *e-verified*, and hence $[\tau] \subseteq \text{int}([F_e])$, we will move the construction of $S_{r(\sigma)}$ to the cone above τ . In this case we say that \mathcal{S}_σ has outcome **fin**. \mathcal{S}_σ 's initial status is ∞ .

Suppose we are at stage $s + 1$ and requirement \mathcal{S}_σ gets activated.

- Suppose first that the last time we visited \mathcal{S}_σ it had status ∞ .

If every $\tau \in S_{f(\sigma)}[s]$ is *e-verified*, then keep the status ∞ and move on to the next requirements $\mathcal{T}_{\sigma \smallfrown 0}$ and $\mathcal{T}_{\sigma \smallfrown 1}$.

Otherwise, let τ_0 be the first $\tau \in S_{f(\sigma)}[s]$ that has not been *e-verified* (first in some ordering of $2^{<\omega}$). Let τ_1 be an extension of τ_0 in $[S][s]$. The plan now is to wait until τ_1 (actually $\tau_1 \smallfrown 0$) is *e-verified*, which will imply that τ_0 is *e-verified*. While we wait, we move the construction of $[S_{f(\sigma)}]$ above τ_1 . Define $r(\sigma)[s + 1] = \tau_1 \smallfrown 0$ (so it has length $s + 1$) and set the status of \mathcal{S}_σ at this stage to **fin**. If $\tau_1 \smallfrown 0$ is never *e-verified*, then we will have $[S_{r(\sigma)}] \subseteq [r(\sigma)] \subseteq \text{int}(F_e)$ as wanted. If at some later stage it is *e-verified*, we redefine $r(\sigma) = f(\sigma)$ and we forget we ever moved $r(\sigma)$ to $\tau_1 \smallfrown 0$. The requirements of lower priority than σ will then continue the work they were doing when they were assuming \mathcal{S}_σ had outcome ∞ and $r(\sigma) = f(\sigma)$. For now, while \mathcal{S}_σ has outcome **fin**, we cannot kill what we were doing above $f(\sigma)$ while we were believing that the the output of \mathcal{S}_σ is ∞ . So, at every stage $t > s$, while the outcome of \mathcal{S}_σ is still **fin**, we have to make sure that every node in $[S][s]$ has at least one extension in $[S][t]$. We have to be careful doing this, because if we never come back to outcome ∞ , we will had built some new paths extending $f(\sigma)$ but not $r(\sigma)$. Let $X_\sigma = [S_{f(\sigma)}] \setminus [S_{r(\sigma)}]$. So, we have to make X_σ satisfy conditions (Sfr1)-(Sfr2). We do it as in the previous construction: On top of each $\tau \in [S_{f(\sigma)}][s]$, $\tau \neq \tau_1$, we use Theorem 1.1 to build a tree Q_τ such that if $[T_\sigma]$ is perfect, then $[Q_\tau]$ is perfect and thin, and $[Q_\tau]$ is finite otherwise. Of course, the construction of Q_τ is done step by step every time \mathcal{S}_σ is active and while it has outcome **fin**.

- Suppose now that the last time \mathcal{S}_σ was active, it had status **fin**. That means that we are waiting for $r(\sigma)$ to get *e-verified*.
 - If $r(\sigma)$ is still not *e-verified*, we keep the status **fin** and we activate the next requirements $\mathcal{T}_{\sigma \smallfrown 0}$ and $\mathcal{T}_{\sigma \smallfrown 1}$. We also do one more step in the construction each of the Q_τ that we started the last time we changed \mathcal{S}_σ 's status from ∞ to **fin**. One thing to notice here is that even if \mathcal{S}_σ stays in status **fin** for ever, it might not be active at every stage. The reason is that there might be some stronger requirement \mathcal{S}_π , $\pi \subset \sigma$, that has outcome ∞ . Even though we are assuming \mathcal{S}_σ knows this is \mathcal{S}_π 's final outcome, \mathcal{S}_π is going to change its status infinitely often to **fin**. Every time it does it, \mathcal{S}_σ gets paralyzed, and when \mathcal{S}_π 's status comes back to ∞ and \mathcal{S}_σ becomes active again, it will find

that some of the paths it was constructing had been extended to longer paths, though no path had been killed. This does not affect \mathcal{S}_σ at all. So long as \mathcal{S}_σ gets to do a new step in the construction of the trees Q_τ infinitely often, it will manage to construct them satisfying (Sfr1)-(Sfr2).

- Suppose now that $r(\sigma)$ has been e -verified since the last time \mathcal{S}_σ was active. Change \mathcal{S}_σ 's status to ∞ . Define $r(\sigma)[s+1] = f(\sigma)$. Let s_0 be the last stage when \mathcal{S}_σ 's status was ∞ . Each string in $[S_{f(\sigma)}]_{[s_0]}$ has at least one extension in $[S_{f(\sigma)}]_{[s]}$; choose one. Kill all the other strings in $[S_{f(\sigma)}]_{[s]}$ by not extending them in $[S_{f(\sigma)}]_{[s+1]}$. The rest of $[S_{f(\sigma)}]_{[s+1]}$ will be defined at the end of stage $s+1$ by other requirements. If a string in $[S_{f(\sigma)}]_{[s_0]}$ was of the form $r(\pi)[s_0]$ for some $\pi \supset \sigma$, redefine $r(\pi)$ to be the chosen extension of it in $[S_{f(\sigma)}]_{[s]}$. Activate requirements $\mathcal{T}_{\sigma \smallfrown 0}$ and $\mathcal{T}_{\sigma \smallfrown 1}$.

3.2. Organization of the construction. Since this construction is an infinite injury one, we will do it on a tree of strategies. The way we do this is very standard, except for the fact that the requirements are not linearly ordered by priority. We could order the requirements linearly and define the tree of strategies the usual way, but instead we continue the style of the previous constructions.

Let TS , the *tree of strategies*, be the set of pairs $\langle \sigma, \alpha \rangle$ where $\sigma \in T$, and α contains beliefs of possible outcomes of the requirements stronger than \mathcal{R}_σ , that is, $\alpha \in \{\mathbf{wai}, \mathbf{sat}, \mathbf{iso}, \mathbf{niso}, \infty, \mathbf{fin}\}^{<\omega}$ satisfies that $\alpha(3e) \in \{\mathbf{wai}, \mathbf{sat}\}$, $\alpha(3e+1) \in \{\mathbf{iso}, \mathbf{niso}\}$, $\alpha(3e+2) \in \{\infty, \mathbf{fin}\}$, and $|\alpha| = |\sigma|$. So, for $i < |\sigma|$, we think of $\alpha(i)$ as the outcome of the requirement at $\sigma \upharpoonright i$, and there is no belief about the outcome of σ . Each $\langle \sigma, \alpha \rangle \in TS$ has a requirement $\mathcal{R}_{\langle \sigma, \alpha \rangle}$ assigned, where \mathcal{R} can be either \mathcal{T} , \mathcal{F} or \mathcal{S} depending on whether $|\sigma|$ is of the form $3e$, $3+1$ or $3e+2$. The outcomes are ordered by $\mathbf{sat} <_L \mathbf{wai}$, $\mathbf{iso} <_L \mathbf{niso}$ and $\infty <_L \mathbf{fin}$. This induces an ordering on TS as follows: $\langle \sigma_0, \alpha_0 \rangle <_L \langle \sigma_1, \alpha_1 \rangle$ if there exists a i such that $\sigma_0 \upharpoonright i+1 = \sigma_1 \upharpoonright i+1$, $\alpha_0 \upharpoonright i = \alpha_1 \upharpoonright i$ and $\alpha_0(i) <_L \alpha_1(i)$. We give $\mathcal{R}_{\langle \sigma_0, \alpha_0 \rangle}$ a *stronger priority* than $\mathcal{R}_{\langle \sigma_1, \alpha_1 \rangle}$ if either $\langle \sigma_0, \alpha_0 \rangle <_L \langle \sigma_1, \alpha_1 \rangle$ or $\langle \sigma_0, \alpha_0 \rangle \subset \langle \sigma_1, \alpha_1 \rangle$. Note that, for example, none of the requirements $\mathcal{R}_{\langle \sigma \smallfrown 0, \alpha \rangle}$ and $\mathcal{R}_{\langle \sigma \smallfrown 1, \alpha \rangle}$ is stronger than the other one.

At every stage s there will be a finite tree $D_s \subset T$ of nodes that get visited and a function $o_s: D_s \rightarrow \{\mathbf{wai}, \mathbf{sat}, \mathbf{iso}, \mathbf{niso}, \infty, \mathbf{fin}\}$ of outcomes. For every $\sigma \in D_s$, the requirement $\mathcal{R}_{\langle \sigma, o_s \upharpoonright \sigma \rangle}$ is activated at stage s , and has outcome, or status, $o_s(\sigma)$, where $o_s \upharpoonright \sigma = \langle o_s(\sigma \upharpoonright i) : i = 0, \dots, |\sigma| - 1 \rangle$. The *true path* $\text{TP}: T \rightarrow \{\mathbf{wai}, \mathbf{sat}, \mathbf{iso}, \mathbf{niso}, \infty, \mathbf{fin}\}$ is defined as usual:

$$\text{TP}(\sigma) = \liminf_{s: \text{TP} \upharpoonright \sigma = o_s \upharpoonright \sigma} o_s(\sigma).$$

Since o_s takes finitely many values, TP exists, and for every $\sigma \in T$, $\mathcal{R}_{\langle \sigma, \text{TP} \upharpoonright \sigma \rangle}$ is activated infinitely often, and requirements to its left (i.e. $<_L$ -less than it) are activated only finitely often.

We define a computable tree $S \subseteq 2^{<\omega}$ and a two partial functions $f, r: TS \rightarrow S$. At the end of the construction we define, for $\sigma \in T$, $f(\sigma) = \lim_{s: \text{TP} \upharpoonright \sigma = o_s \upharpoonright \sigma} f(\langle \sigma, \text{TP} \upharpoonright \sigma \rangle)[s]$ and $r(\sigma) = \lim_{s: \text{TP} \upharpoonright \sigma = o_s \upharpoonright \sigma} r(\langle \sigma, \text{TP} \upharpoonright \sigma \rangle)[s]$, where $o_s \upharpoonright \sigma = \langle o_s(\sigma \upharpoonright i) : i = 0, \dots, |\sigma| \rangle$.

E-mail address: Rod.Downey@mcs.vuw.ac.nz

URL: <http://www.mcs.vuw.ac.nz/~downey>

E-mail address: antonio@mcs.vuw.ac.nz

URL: www.math.uchicago.edu/~antonio

SCHOOL OF MATHEMATICS, STATISTICS AND COMPUTER SCIENCE, VICTORIA UNIVERSITY, P.O.
BOX 600, WELLINGTON, NEW ZEALAND