Topics in Algorithmic Randomness and Computability Theory

by

Michael Patrick McInerney

A thesis
submitted to the Victoria University of Wellington
in fulfilment of the
requirements for the degree of
Doctor of Philosophy
in Mathematics.

Victoria University of Wellington
2016

Abstract

This thesis establishes results in several different areas of computability theory.

The first chapter is concerned with algorithmic randomness. A well-known approach to the definition of a random infinite binary sequence is via effective betting strategies. A betting strategy is called integer-valued if it can bet only in integer amounts. We consider integer-valued random sets, which are infinite binary sequences such that no effective integer-valued betting strategy wins arbitrarily much money betting on the bits of the sequence. This is a notion that is much weaker than those normally considered in algorithmic randomness. It is sufficiently weak to allow interesting interactions with topics from classical computability theory, such as genericity and the computably enumerable degrees. We investigate the computational power of the integer-valued random sets in terms of standard notions from computability theory.

In the second chapter we extend the technique of forcing with bushy trees. We use this to construct an increasing ω-sequence $\langle \boldsymbol{a}_n \rangle$ of Turing degrees which forms an initial segment of the Turing degrees, and such that each $\boldsymbol{a}_{n+1}$ is diagonally noncomputable relative to $\boldsymbol{a}_n$. This shows that the DNR_0 principle of reverse mathematics does not imply the existence of Turing incomparable degrees.

In the final chapter, we introduce a new notion of genericity which we call ω-change genericity. This lies in between the well-studied notions of 1- and 2-genericity. We give several results about the computational power required to compute these generics, as well as other results which compare and contrast their behaviour with that of 1-generics.

Acknowledgments

I would like to thank my supervisors Rod Downey and Noam Greenberg for their guidance. I would also like to thank my other coauthors Mingzhong Cai and George Barmpalias, and Dan Turetsky and Sasha Melnikov for helpful comments and discussions.

Contents

Chapter 1

Introduction

This thesis is divided into three parts, each in an area of computability theory.

The first part, which is joint work with George Barmpalias and Rod Downey, concerns algorithmic randomness, which uses ideas from computability theory to give a mathematically rigorous definition of a random binary sequence.

Early approaches to the definition of a random binary sequence were concerned with statistical properties. One such property is the law of large numbers. Suppose that we repeatedly flipped a fair coin. We would expect that the ratio of heads to tails would tend to 1 as the number of coin flips increased. The sequence 01010101... obeys the law of large numbers, but is certainly not random according to our intuition. Von Mises ([45]) suggested that a sequence should be considered random if it obeys the law of large numbers, and furthermore that certain subsequences obey the law of large numbers. He did not however specify which subsequences should be considered.

The first satisfactory definition of a random binary sequence was given by Martin-Löf ([36]) using ideas from computability theory. The statistical tests here are certain kinds of computable sets of measure zero. Since then, many variations on this idea have produced a range of randomness notions.

We might also consider a binary sequence to be random if it is unpredictable. Suppose that we begin with some amount of money, and we bet on the bits of the sequence. The sequence is revealed to us one bit at a time, and we bet on what the next bit will be. If the sequence is random, then we should not be able to win arbitrarily much money in this way. The idea of a betting strategy can be formalised with the notion of a *martingale*. Schnorr ([41]) effectivised the notion of a martingale, and showed that this approach was equivalent to the approach using statistical tests that had already been developed.

The martingale approach is especially useful in considering randomness notions that are weaker than Martin-Löf's. The particular type of martingale associated with Martin-Löf randoms is quite complicated. The amount that we bet on a particular finite binary sequence can be a *left-c.e.* real number. This number is given by enumerating, in some computable way, the set of rational numbers less than it. In particular, this may be an infinite process. If we instead require that the number be a *computable* number (that is, there is an algorithm that on input n, returns the nth bit in the binary expansion), we arrive at the notion of a *computably random* sequence.

We can further weaken this notion by restricting the kind of bets we can make. The weakest such notion considered so far is known as *integer-valued randomness*. Here, we can only bet in integer amounts. This was introduced by Bienvenu, Stephan, and Teutsch in [5], where its interactions with all other commonly studied randomness notions were described. In Chapter 2, we establish several results relating integer-valued randomness to classical measures of computational power from classical computability theory. Integer-valued randomness seems sufficiently weak for there to be significant interaction in ways that have not been observed before with stronger notions of randomness. In particular, we establish connections between integer-valued randomness and genericity (which is the subject of Chapter 4), and obtain several results about integer-valued randomness and the computably enumerable degrees, which are certainly the most well-studied objects in classical computability theory.

The second part of the thesis, which is joint work with Mingzhong Cai and Noam Greenberg, concerns a technique known as forcing with bushy trees. This particular technique has been used in the last few years to exhibit sets with interesting computational properties.

As is standard, our abstract model of computation is the Turing machine. A set of natural numbers A is said to be *Turing reducible* to a set of natural numbers B if there is a Turing machine, which when equipped with information about membership in B, can decide membership in A. We also say that B *computes* A. Two sets A and

B are said to be *Turing equivalent* if each is Turing reducible to the other. This gives an equivalence relation on the set of all subsets of natural numbers. The equivalence classes are called *Turing degrees*. We then say that a Turing degree $\boldsymbol{a}$ is Turing reducible to a Turing degree $\boldsymbol{b}$ if for some $A \in \boldsymbol{a}$ and $B \in \boldsymbol{b}$, A is Turing reducible to B. Turing degrees are designed to capture *information content*; all sets with the same information are in the same Turing degree.

The Turing degree $\mathbf{0}$ is the degree containing the empty set $\emptyset$. We consider the sets in $\mathbf{0}$ to contain no information, since membership in such a set can be easily determined using a Turing machine. A Turing degree $\boldsymbol{a}$ is *minimal* if it is not equal to $\mathbf{0}$, and the only Turing degrees reducible to $\boldsymbol{a}$ are $\mathbf{0}$ and $\boldsymbol{a}$ itself. Then $\boldsymbol{a}$ is minimal in the sense that if it contained any less information, it would be $\mathbf{0}$. Of course, these degrees are considered computationally weak.

There are many ways in which we may consider a Turing degree to be computationally strong. One such way is if it is able to compute a function which is so called *diagonally noncomputable* (whose definition we give in Chapter 3). One question is whether a Turing degree can be both minimal, and compute a diagonally noncomputable function. This was answered affirmatively by Kumabe and presented in [32], using forcing with bushy trees. In Chapter 3 we extend this result.

This result has an application in reverse mathematics, which is a programme in the foundations of mathematics which asks, given a theorem ordinary classical mathematics, which set existence axioms are necessary in its proof. One system, DNR_0, ensures the existence of a diagonally noncomputable function. A slightly stronger system, $WWKL_0$, ensures the existence of a Martin-Löf random set. Conidis ([15]) showed that the system $WWKL_0$ implies Turing incomparability. That is, in every collection of sets satisfying $WWKL_0$, there are sets A and B such that neither is Turing reducible to the other. However, our result shows that DNR_0 does not imply Turing incomparability. In fact, it does not imply the existence of a pair of Turing incomparable sets.

The final part of the thesis concerns genericity. We may consider random sets as *typical*, in that almost all sets, in the sense of measure, are random. If instead we look at what it means for a set to be typical with respect to category, we arrive at the generic sets; the generic sets are comeager in the set of all subsets of natural numbers. In [26], Jockusch introduced restricted forms of genericity. For every $n \in \mathbb{N}$, we have the n-generic sets. These form a proper hierarchy: every $n+1$-generic set is n-generic, but the converse does not hold.

The greater the n, the more typical we consider an n-generic set to be. In many cases, typical behaviour starts with the 2-generic sets, and will fail for 1-generic sets. As an example, the collection of sets computing a 2-generic set has measure 0, whereas the collection of sets computing a 1-generic set has measure 1. Thus it is of great interest to determine exactly when typical behaviour starts, and we may use notions of genericity intermediate between 1- and

2-genericity to more finely specify this.

Several such notions have already been defined. The most well-known is *pb-genericity*, which was introduced by Downey, Jockusch, and Stob in [20]. To highlight the difference, we consider what we must do in order to construct generic sets. In computability theory, we often construct a set with a certain property by satisfying infinitely many *requirements*. To construct a 1-generic set, we need to act only once to satisfy each requirement. To construct a pb-generic set, we may need to act many times to satisfy a requirement, but the number of times we must act is known to us before the construction begins.

In Chapter 4 I introduce a new notion of genericity intermediate between 1- and 2-genericity, which I call ω-change genericity. Here, in order to construct an ω-change generic, we may need to act many times, but the number of times is only revealed to us during the course of the construction. This introduces a more dynamic flavour to the constructions. I establish several results which quantify the level of computational power required to compute an ω-change generic set. These are related to a hierarchy recently introduced by Downey and Greenberg in [17]. I also extend a result of Chong and Downey from [13] to give a characterisation of those sets which are computable in an ω-change generic. I finally comment on the downward density of ω-generics below $\mathbf{0}'$.

Chapter 2

Integer-valued randomness

This chapter is joint work with George Barmpalias and Rod Downey and has appeared in [3].

2.1 Introduction

An interesting strategy for someone who wishes to make a profit by betting on the outcomes of a series of unbiased coin tosses, is to double the the amount he bets each time he places a bet. Then, independently of whether he bets on heads or tails, if the coin is fair (i.e. the sequence of binary outcomes is random) he is guaranteed to win infinitely many bets. Furthermore, each time he wins he recovers all previous losses, plus he wins a profit equal to the original stake. This is a simple example from a class of betting strategies that originated from, and were popular in 18th century France. They are known as *martingales*. The "success" of this strategy is essentially equivalent to the fact that a symmetric one-dimensional random walk will eventually travel an arbitrarily long distance to the right of the starting point (as well as an arbitrarily long distance to the left of the starting point).

So what is the catch? For such a strategy to be maintained, the player needs to be able to withstand arbitrarily large losses, and such a requirement is not practically feasible. In terms of the random walk, this corresponds to the fact that, before it travels a large distance to the right of the starting point, it is likely to have travelled a considerable distance to the left of it.

2.1.1 Martingales and randomness

Martingales have been reincarnated in probability theory (largely though the work of Doob), as (memoryless) stochastic processes (Z_n) such that the conditional expectation of each Z_{n+1} given Z_n remains equal to the expectation of Z_0. The above observations on a fair coin-tossing game are now theorems in the theory of martingales. For example, Doob's maximal martingale inequality says that with probability 1, a non-negative martingale is bounded. Intuitively this means that, if someone is not able (or willing) to take credit (so that he continues to bet after his balance is negative) then the probability that he makes an arbitrarily large amount in profit is 0.

Martingales in probability rest on a concept of randomness in order to determine (e.g. with high probability) or explain the outcomes of stochastic processes. In turns out that this methodology can be turned upside down, so that certain processes are used in order to define or explain the concept of randomness. Such an approach was initiated by Schnorr in [41], and turned out to be one of the standard and most intuitive methods of assigning meaning to the concept of randomness for an individual string or a real (i.e. an infinite binary sequence, a point in the Cantor space). This approach is often known as the *unpredictability paradigm*, and it says that it should not be possible for a computable predictor to be able to predict bit $n+1$ of a real X based on knowledge of bits $1, \dots, n$ of X, namely $X \upharpoonright n$. The unpredictability paradigm can be formalized by using *martingales*, which (for our purposes) can be seen as betting strategies. We may define a martingale to be a function $f : 2^{<\omega} \to \mathbb{R}^{\geqslant 0}$ which obeys the following fairness condition:

$$f(\sigma) = \frac{f(\sigma 0) + f(\sigma 1)}{2}.$$

If f is partial, but its domain is downward closed with respect to the prefix relation on finite strings, then we say that f is a *partial martingale*.

In probability terms, f can be seen as a stochastic process (a series of dependent variables) Z_s where Z_s represents the capital of a player at the end of the sth bet (where there is 50% chance for head or tails). Then the fairness condition says that the expectation of f at stage $s+1$ is the same as the value of f at stage s. In other words, the fairness condition says that the expected growth of f at each stage of this game is 0. If we interpret f as the capital of a player who bets on the outcomes of the coin tosses, the fairness condition says that there is no bias in this game toward the player or the house. Moreover note that our definition of a martingale as a betting strategy requires that it is non-negative. Recalling our previous discussion about gambling systems, this means that we do not allow the player to have a negative balance. This choice in the definition is essential, as it prevents the success of a 'martingale betting system' as we described it. Continuing with our definition of martingales as betting strategies, we say that f succeeds on a real X if

$$\limsup_{n\to\infty} f(X \upharpoonright n) = \infty.$$

Schnorr [41] was interested in an algorithmic concept of randomness. Incidentally, Martin-Löf [36] had already provided a mathematical definition of randomness based on computability theory and effective measure theory. But Schnorr wanted to approach this challenge via the intuitive concept of betting strategies. He proved that a real (i.e. an infinite binary sequence) X is Martin-Löf random if and only if no effective martingale can succeed on it. Here "effective" means that f is computably approximable from below. Schnorr's result is an effective version of the maximal inequality for martingales in probability theory, which says that with probability 1 a non-negative martingale is bounded. There is a huge literature about the relationship between martingales and effective randomness, and variations on the theme, such as computable martingales and randomness, partial computable martingales, nonmonotonic martingales, polynomial time martingales, etc. We refer the reader to Downey and Hirschfeldt [18] and Nies [37] for some details and further background.

2.1.2 Why integer-valued martingales?

Recall the standard criticism of martingale betting systems, i.e. that their success depends on the ability of the player to sustain arbitrarily large losses. This criticism lead (for the purpose of founding algorithmic randomness) to defining a martingale as a function from the space of coin-tosses to the non-negative reals (instead of all the reals) which represent the possible values of the capital available to the player. There is another criticism on such betting strategies that was not taken into account in the formal definition. Schnorr's definition of a martingale (as a betting strategy) allows betting infinitesimal (i.e. arbitrarily small) amounts. Clearly such an option is not available in real gambling situations, say at a casino, where you cannot bet arbitrarily small amounts on some outcome. It becomes evident that restricting the betting strategies to a discrete range results in a more realistic concept of betting. Such considerations led Bienvenu, Stephan and Teutsch [5] to introduce and study *integer-valued martingales*, and the corresponding randomness notions. Interestingly, it turns out that the algorithmic randomness based on integer-valued martingales is quite different from the theory of randomness based on Martin-Löf [36] or Schnorr [41] (as developed in the last 30 years, see [18, 37] for an overview). The reason for this difference is that most of the classical martingale arguments in algorithmic randomness make substantial use of the property of being able to bet infinitesimal amounts (thereby effectively avoiding bankruptcy at any finite stage of the process).

Quite aside from the motivations of examining the concept of integer-valued martingales for its own sake, if we are to examine the randomness that occurs in practice, then such discretised randomness will be the kind we would get. The reason is evident: we can only use a finite number of rationals for our bets, and these scale to give integer values. *Additionally*, at the more speculative level, if the universe is granular, finite, and not a manifold, then if there is any randomness to be had (such as in quantum mechanics) it will be integer-valued for the same reason.

2.1.3 Integer randomness notions and computability

We formally introduce and discuss the notions of integer-valued randomness in the context of computability theory. For the purposes of narrative flow, we will assume that the reader is familiar with the basics of algorithmic randomness. Schnorr based algorithmic randomness on the concept of effective strategies. Along with this foundational work, he introduced and philosophically argued for a randomness notion which is weaker than Martin-Löf randomness and is now known as *Schnorr randomness*. Further notions, like *computable randomness*, are quite natural from the point of view of betting strategies and have been investigated extensively (see for example Chapter 7 of [18] and Chapter 7 of [37]). Integer-valued martingales induce randomness notions with properties that are quite a different in flavour from those of, for instance, Martin-Löf randomness, computable randomness and the like. Our goal in the present chapter is to clarify the relationship between integer-valued randomness and classical degree classes which measure levels of computational power.

Definition 2.1.1 (Integer-valued martingales)**.** Given a finite set $F \subseteq \mathbb{N}$, we say that a martingale f is F-valued if $f(\sigma i) = f(\sigma) \pm k$ for some $k \in F$. A martingale is integer-valued if it is $\mathbb{N}$-valued, and is single-valued if $F = \{a, 0\}$ for some $a \neq 0$.

Note that a martingale is F-valued if at any stage we can only bet k dollars for some $k \in F$ on one of the outcomes $i \in \{0, 1\}$, and must lose k dollars if $1 - i$ is the next bit. We note that partial integer-valued martingales are defined as in Definition 2.1.1, only that the martingales can be partial. In the following we often say that, given a string σ, the string $\sigma 0$ is the *sibling* of $\sigma 1$ (and $\sigma 1$ is the *sibling* of $\sigma 0$).

If we restrict our attention to the countable class of computable or partial computable martingales, we obtain a number of algorithmic randomness notions. For example, a real is [partial] computably random if no [partial] computable martingale succeeds on it. Similar notions are obtained if we consider integer-valued martingales.

Definition 2.1.2 (Integer-valued randomness)**.** A real X is [partial] integer-valued random if no [partial] computable integer-valued martingale succeeds on it. Moreover X is finitely-valued random if for each finite set $F \subseteq \mathbb{N}$, no computable F-valued martingale succeeds on it, and is single-valued random if no computable single-valued martingale succeeds on it.

Integer-valued	Computable
Finite-valued	Single-valued
Partial integer-valued	Partial computable

Table 2.1: Randomness notions based on effective martingales

We list these randomness notions in Table 2.1, along with the traditional randomness notions computable and partial computable randomness. Note that partial integer-valued randomness is stronger than integer-valued randomness, just as partial computable randomness is stronger than computable randomness. Bienvenu, Stephan and Teutsch [5] clarified the relationship between integer-valued, single-valued and a number of other natural randomness notions. Figure 2.1 illustrates some of the implications that they obtained. We already know (see [18]) that computable randomness implies Schnorr randomness, which in turn implies Kurtz randomness (with no reversals) and that Schnorr randomness implies the law of large numbers. Bienvenu, Stephan and Teutsch proved that if we add the above notions to the diagram in Figure 2.1, no other implications hold apart from the ones in the diagram and the ones just mentioned. In addition, we may add a node for 'partial computably random' and an arrow from it leading to the node 'computably random'. Nies showed in [37, Theorem 7.5.7] that the converse implication does not hold, i.e. there are computably random reals which are not partial computably random. A strong version of this fact holds for integer-

$$\begin{tikzcd}
\text{computably random} \arrow[r] & \text{integer-valued random} \arrow[d] & \text{Kurtz random} \arrow[d] \\
\text{bi-immune} & \text{finitely-valued random} \arrow[l] \arrow[r] & \text{single-valued random}
\end{tikzcd}$$

Figure 2.1: Implications between randomness notions obtained in [5].

valued randomness. We show that there are integer-valued random reals which not only are not partial integer-valued random, but they do not contain any partial integer-valued random reals in their Turing degree.

One interesting observation of Bienvenu, Stephan and Teutsch [5], was that integer-valued randomness was a meeting point of genericity (and hence category) and measure since weakly 2-generic sets are integer-valued random. Hence the integer-valued randoms are co-meager as well as having measure 1. The reader should recall that a subset of $\mathbb{N}$ is called n-generic if it meets or avoids all Σ^0_n sets of strings, and is weakly n-generic if it meets all dense Σ^0_n sets of strings. The reason that highly generic reals are integer-valued random is that there is a finitary strategy to make a set integer-valued random, since we can force an opponent who bets to lose. The point here is that if the minimum bet is one dollar and he has k dollars to spend, then he can lose at most k times, so we can use a finite strategy to force the opponent into a cone (in Cantor space) where he cannot win. This finite strategy is not available if arbitrarily small bets are allowed. Naturally the question arises as to what level of genericity is needed for constructing integer-valued randoms. Bienvenu, Stephan and Teutsch [5] proved that it is possible to have a 1-generic which is *not* integer-valued random. So the answer they gave to this question is ‘somewhere between weak 2 and 1-genericity’. As we see in the next section, we give a more precise answer.

2.1.4 Our results, in context

Bienvenu, Stephan and Teutsch [5] showed that the class of integer-valued random is co-meager, so sufficient genericity is a guarantee for this kind of randomness. They also quantified this statement via the hierarchy of genericity, showing that the genericity required lies somewhere between weak 2-genericity and 1-genericity. We show that a notion of genericity from [20] which is known as *pb*-genericity, implies (partial) integer-valued randomness. Recall from [20] that set of strings S is *pb*-dense if it contains the range of a function $f : 2^{<\omega} \to 2^{<\omega}$ with a computable approximation (f_s) such that $f(\sigma) \succcurlyeq \sigma$ for all strings σ and $|\{s \mid f_{s+1}(\sigma) \neq f_s(\sigma)\}| \leqslant h(\sigma)$ where the function $h : 2^{<\omega} \to \omega$ is primitive recursive. A real X is *pb*-generic if every *pb*-dense set of strings contains a prefix of X.

Theorem 2.1.3 (Genericity for integer-valued randoms)**.** *Every pb-generic real is (partial) integer-valued random.*

This result might suggest that integer-valued randomness and partial integer-valued randomness are not easily distinguishable. In Section 2.2.2 we present a rather elaborate finite injury construction of a real which is integer-valued random but not partial integer-valued random. In Section 2.2.3 this construction is modified into a $0''$ tree argument, which proves the following degree separation between the two randomness notions.

Theorem 2.1.4 (Degree separation of randomness notions)**.** *There exists an integer-valued random $X <_T \mathbf{0}'$ which does not compute any partial integer-valued random.*

We are interested in classifying the computational power that is associated with integer-valued randomness. Computational power is often represented by properties of degrees, which in turn define classes like the degrees which can solve the halting problem, or the array non-computable degrees from [20]. The reader might recall that A is array noncomputable if and only if for all $f \leqslant_{wtt} \varnothing'$, there is a function $g \leqslant_T A$ such that $g(n) > f(n)$ for infinitely many n. This class has turned out to be ubiquitous and characterized classes defined by many distinct combinatorial properties. Recall that a presentation of a real A is a c.e. prefix-free set of strings representing an open set of Lebesgue measure α. The c.e. array noncomputable degrees are exactly the c.e. degrees that

(a) contain c.e. sets A of infinitely often maximal (i.e. $2\log n$) Kolmogorov complexity; (Kummer [33])

(b) have effective packing dimension 1; (Downey and Greenberg [16])

(c) compute left-c.e. reals $B <_T A$ such that every presentation of A is computable from B; (Downey and Greenberg [17])

(d) compute a pair of disjoint c.e. sets such that every separating set for this pair computes the halting problem; (Downey, Jockusch, and Stob [19])

(e) do not have strong minimal covers; (Ishmukhametov [25])

Also by Cholak, Coles, Downey, Herrmann [11] the array noncomputable c.e. degrees form an invariant class for the lattice of Π^0_1 classes via the thin perfect classes.

Theorem 2.1.3 can be used to show that a large class of degrees compute (partial) integer-valued randoms. By [20], every array noncomputable degree computes a *pb*-generic. Therefore Theorem 2.1.3 has the following consequence.

Corollary 2.1.5 (Computing integer-valued randoms)**.** *Every array noncomputable degree computes a (partial) integer-valued random.*

Note however that an integer-valued random need not be of array noncomputable degree. Indeed, it is well known that there are array computable Martin-Löf randoms. A converse of Corollary 2.1.5 can be obtained for the c.e. degrees, as Theorem 2.1.7 shows.

While genericity is an effective tool for exhibiting integer-valued randomness in the global structure of the degrees (as we demonstrated above) it is incompatible with computable enumerability. Since generic degrees (even 1-generic) are not c.e., investigating integer-valued randomness in the c.e. degrees requires a different analysis. Already the fact that randomness can be exhibited in the c.e. degrees is quite a remarkable phenomenon, and restricted to weak versions of randomness. Martin-Löf randomness is the strongest standard randomness notion that can be found in the c.e. degrees. A c.e. degree contains a Martin-Löf random set if and only if it is complete (i.e. it is the degree of the halting problem). Furthermore, the complete c.e. degree contains the most well-known random sequence—Chaitin's Ω—which is the measure of the domain of a universal prefix-free machine, the universal *halting probability*. An interesting characteristic of this random number is that it is left-c.e., i.e. it can be approximated by a computable increasing sequence of rationals. Weaker forms of randomness—like computable and Schnorr randomness—can be found in incomplete c.e. degrees, even in the form of left-c.e. sets. For example, Nies, Stephan and Terwijn [38] showed that the c.e. degrees which contain computably and Schnorr random sets are exactly the high c.e. degrees. Moreover each high c.e. degree contains a computably random left-c.e. set and a Schnorr random left-c.e. set. We prove an analogous result for integer-valued randomness and partial integer-valued randomness.

Theorem 2.1.6 (C.e. degrees containing integer-valued random left-c.e. sets)**.** *A c.e. degree contains a (partial) left-c.e. integer-valued random if and only if it is high.*

This result is pleasing, but this is where the similarities between computable randomness (based on computable betting strategies) and integer-valued randomness (based on integer-valued computable betting strategies) end, at least with respect to the c.e. degrees. We note that

> In the c.e. degrees, the following classes are equal to the high degrees:
>
> (i) degrees containing computably random sets;
>
> (ii) degrees containing left-c.e. computably random sets;
>
> (iii) degrees computing computably random sets.
>
> (2.1)

This characterization follows from the following facts, where (a) is by Schnorr [41], (c) was first observed by Kučera [30], and (b), (d) are from [38].

(a) computable randomness implies Schnorr randomness;

(b) a Schnorr random which does not have high degree is Martin-Löf random;

(c) a Martin-Löf random of c.e. degree is complete;

(d) every high c.e. degree contains a computably random left-c.e. set.

In the case of integer-valued randomness (2.1) fails significantly. In particular, the c.e. degrees that compute integer-valued randoms are not the same as the c.e. degrees that contain integer-valued randoms. In fact, we provide the following characterization of the c.e. degrees that compute integer-valued randoms.

Theorem 2.1.7 (C.e. degrees computing integer-valued randoms)**.** *A c.e. degree computes an integer-valued random if and only if it is array noncomputable.*

In view of this result it is tempting to think that that every c.e. array noncomputable might contain an integer-valued random. We will see however in Section 2.5.1 that there are array noncomputable c.e. degrees which do not contain integer-valued randoms. In fact, Theorem 2.1.8 is an extreme version of this fact, which is tight with respect to the jump hierarchy.

We have seen that high c.e. degrees are powerful enough to contain integer-valued randoms, even left-c.e. integer-valued randoms. However, while we know that left-c.e. integer-valued randoms necessarily have high degree, the question arises as to whether a weaker jump class is sufficient to guarantee that a c.e. degree contains an integer-valued random, if we no longer require that the set is left-c.e. We give a negative answer to this question by the following result, which we prove in Section 2.5.2 by a $0'''$-argument.

Theorem 2.1.8. *There is a high$_2$ c.e. degree which does not contain any integer-valued randoms.*

Note that this result shows the existence of array noncomputable c.e. degrees which do not contain integer-valued randoms, in stark contrast to Theorem 2.1.7.

Furthermore, the c.e. degrees that contain integer-valued randoms are not the same as the c.e. degrees that contain left-c.e. integer-valued randoms. In fact, in contrast with Theorem 2.1.6, there exists a low c.e. degree containing an integer-valued random set. More generally, we can find c.e. degrees containing integer-valued random sets in every jump class. Section 2.4 is devoted to the proof of this result.

Theorem 2.1.9 (Jump inversion for c.e. integer-valued random degrees)**.** *If c is c.e. in and above $0'$ then there is an integer-valued random A of c.e. degree with $A' \in c$.*

There are a number of open questions and research directions pointed by the work in this chapter. For example, is there a c.e. degree which contains an integer-valued random but does not contain any partial integer-valued randoms? More generally, which degrees contain partial computable randoms? Algorithmic randomness based on partial martingales is a notion that remains to be explored on a deeper level.

2.2 Genericity and partial integer-valued randoms

Bienvenu, Stephan, and Teutsch [5, Theorem 8] showed that every weakly 2-generic set is integer-valued random. In Section 2.2.1 we give a proof of Theorem 2.1.3, i.e. that *pb*-genericity is sufficient for (partial) integer-valued randomness.

Hence a certain notion of genericity (*pb*-genericity) is a source of integer-valued randomness. In fact, by Theorem 2.1.3, every *pb*-generic is not only integer-valued random but also partial integer-valued random. We do have concrete examples of reals that are integer-valued random but not partial integer-valued random. Section 2.2.2 is dedicated to constructing such an example. We give the basic construction of a Δ^0_2 real which is integer-valued random but not partial integer-valued random. We give this in full detail, as it is based on an interesting idea.

In Section 2.2.3 we provide the necessary modification of the previous construction in order to show that the degrees of integer-valued randoms and partial integer-valued randoms can also be separated, even inside Δ^0_2. In particular, we are going to prove Theorem 2.1.4, i.e. that there is a Δ^0_2 integer-valued random which does not compute any partial integer-valued randoms. These modifications are essentially the implementation of the main argument of Section 2.2.2 on a tree, which results from the additional requirements that introduce infinitary outcomes that need to be guessed (i.e. the totality of the various functionals with oracle the constructed set). Given the original strategies and construction, the tree argument is fairly standard.

2.2.1 Proof of Theorem 2.1.3

For every partial computable integer-valued martingale m with effective approximation (m_s) we define a function $\hat{m} : 2^{<\omega} \to 2^{<\omega}$ with uniformly computable approximation $(\hat{m}_s)$. Let $\hat{m}_0(\sigma) = \sigma$ for all strings σ. Inductively in s, suppose we have defined $\hat{m}_s$. At stage $s + 1$, if $m_{s+1}(\hat{m}_s(\sigma))$ is defined and there exists an extension τ of it of length $\leqslant s + 1$ such that $m_{s+1}(\tau)$ is defined and $m_{s+1}(\tau) < m_{s+1}(\hat{m}_s(\sigma))$, then we define $\hat{m}_{s+1}(\sigma)$ to be the least such string τ (where strings are ordered first by length and then lexicographically). Otherwise let $\hat{m}_{s+1}(\sigma) = \hat{m}_s(\sigma)$.

Since m is an integer-valued martingale we have $|\{s \mid \hat{m}_{s+1}(\sigma) \neq \hat{m}_s(\sigma)\}\,| \leqslant m(\sigma) + 1$ and $m(\sigma) \leqslant 2^{|\sigma|} \cdot m(\varnothing)$. Note that given any partial computable martingale m, the range of $\hat{m}$ is dense. So every pb-generic intersects the range of $\hat{m}$ for every partial computable martingale m. Moreover given any partial computable martingale m, the range of $\hat{m}$ is a subset of

$$W_m = \{\sigma \mid m(\sigma') \simeq m(\sigma) \text{ for all extensions } \sigma' \text{ of } \sigma\}.$$

So every pb-generic intersects W_m for each partial computable martingale m. This means that every pb-generic is partial integer-valued random.

2.2.2 An integer-valued random which is not partial integer-valued random

It suffices to construct an integer-valued random set $A \leqslant_T \varnothing'$ and a partial integer-valued martingale m which succeeds on A. We will define a computable approximation (A_s) which converges to a set A which has the required properties. Let $\langle n_e \rangle_{e \in \omega}$ be an effective list of all partial computable integer-valued martingales. For each $e \geqslant 2$ we need to satisfy the requirements

R_e**:** If n_e is total, then n_e does not succeed on A.

Q_e**:** m wins (at least) \$$e$ on A.

We first see how we might meet one requirement R_0. We begin by setting $A_0 = 1^\omega$ and defining m to start with \$2 and wager \$1 on every initial segment of A_0. If we later see that n_0 has increased its capital along A_0, then we would like to move our approximation to A so that n_0 decreases in capital. In changing A_s to decrease n_0's capital we may also decrease m's capital along A_s. If n_e had, say, \$10 in capital at some point, then as it loses at least a dollar every time it decreases in capital, it can lose at most 10 times. Our martingale, if it bets in \$1 wagers, can withstand losing \$10 only if its capital at that point is at least \$11.

If m does not have sufficient capital for us to start attacking immediately, we must find a way to increase its capital. We can increase m's capital to \$$k$ as follows. We have not yet defined m on any string extending 0. We therefore wait until n_0 has halted on all strings of length k. If this never happens, then n_0 is not total, and R_0 is met. If n_0 does halt on all strings of length k, we pick a string τ of length k extending 0 such that $n_0(\tau) \leqslant n_0(0)$. Such a string must exist since n_0 is a (partial) martingale. We are then free to define m to wager \$1 on every initial segment of τ and set $A_s = \tau\hat{\ }1^\omega$. We will then have $m(\tau) > n_0(\tau)$. If n_0 later increases its capital along A_s we will be able to change the approximation to A to decrease n_0's capital. As m now has greater capital than n_0, we will be able to decrease n_0's capital to \$0 while ensuring that m does not run out of money.

When dealing with multiple requirements, we must take care in defining m as it is a global object. We set a restraint r_e for every

$e \in \omega$. We arrange things so that only R_e will be able to define m on a string extending $A_s \upharpoonright r_{e,s}\hat{\ }0$. Suppose R_e has not required attention since it was last injured. When we see n_e increase its capital above $n_e(A_s \upharpoonright r_e)$, rather than starting to attack immediately, even if possible, we choose a string τ' extending $A_s \upharpoonright r_{e,s}\hat{\ }0$ and define m such that $m(\tau') - m(A_s \upharpoonright r_{e,s}\hat{\ }0) > n_e(A_s \upharpoonright r_{e,s}\hat{\ }0)$. We injure requirements of weaker priority by lifting their restraints. We may then decrease n_e's capital to \$0 and still have m left with some capital. We now turn to the formal details of the construction.

We have for every requirement R_e a restraint r_e. At every stage a requirement will either be declared to be waiting for convergence at some length, or declared to not be waiting for convergence at any length. As usual, this will stay in effect at the next stage unless otherwise mentioned. We say that R_e requires attention at stage s if either

(i) R_e was declared to not be waiting for convergence at any length at stage s, and there is l such that

   (a) $l > r_{e,s}$,

   (b) $n_{e,s}(\sigma)\downarrow$ for all strings σ of length l, and

   (c) $n_e(A_s \upharpoonright l) > n_e(A_s \upharpoonright (l-1))$, or

(ii) R_e was declared to be waiting for convergence at length h at stage s, and $n_{e,s}(\sigma)\downarrow$ for all strings σ of length h.

In Case (i) we say that R_e requires attention through l. We say that

Q_e requires attention at stage s if $m(A_s \upharpoonright r_{e,s}) < e$. We order the requirements as $R_0, R_1, R_2, Q_2, R_3, Q_3, \ldots$.

Construction

Stage 0: Set $A_0 = 1^\omega$ and $m(\lambda) = 2$, and let m wager \$1 on every initial segment of A_0. Set $r_{e,0} = e$ for all $e \in \omega$. For all $e \in \omega$, declare that R_e is not waiting for convergence at any length at stage 1.

Stage s, $s \geqslant 1$: Find the requirement of strongest priority which requires attention at stage s. (If no such requirement exists, go to the next stage.) There are several cases.

Case 1: R_e requires attention at stage s in Case (i). Has R_e required attention since it was last injured?

Subcase 1a: No. Declare R_e to be waiting for convergence at length $n_e((A_s \upharpoonright r_{e,s})\hat{\ }0) + 1 + (r_{e,s} + 1)$ at stage $s + 1$.

Subcase 1b: Yes. Suppose l is least such that R_e requires attention through l at stage s. Let $A_{s+1} = (A_s \upharpoonright (l-1))\hat{\ }(1 - A_s(l-1))\hat{\ }1^\omega$. Define m to wager \$1 on every initial segment of A_{s+1} of length at least $l + 1$. For all $e' > e$, let $r_{e',s+1}$ be a fresh large number such that for all $e_1 < e_2$ we have $r_{e_1,s+1} < r_{e_2,s+1}$, and declare that $R_{e'}$ is not waiting for convergence at any length at stage $s + 1$.

Case 2: R_e requires attention at stage s in Case (ii). Suppose R_e was declared to be waiting for convergence at length h at stage t.

Choose a string τ above $A_s \upharpoonright r_{e,s}\hat{\ }0$ of length h such that $n_e(\tau) \leqslant n_e(A_s \upharpoonright r_{e,s}\hat{\ }0)$. Define m to wager \$1 along every initial segment of τ with length in $(r_{e,s} + 1, |\tau|]$. Set $A_{s+1} = \tau\hat{\ }1^\omega$. Define m to wager

\$1 on every initial segment of A_{s+1} of length at least $|\tau| + 1$. For all $e' > e$, let $r_{e',s+1}$ be a fresh large number such that if $e_1 < e_2$ then $r_{e_1,s+1} < r_{e_2,s+1}$. For all $e'' \geqslant e$, declare that $R_{e''}$ is not waiting for convergence at any length at stage $s + 1$.

Case 3: Q_e requires attention at stage s. Let τ be the least string extending $A_s \upharpoonright r_{e,s}$ for which $m(\tau) = e$. Set $A_{s+1} = \tau\hat{\ }1^\omega$ and for all $e' > e$, let $r_{e',s+1}$ be a fresh large number such that if $e_1 < e_2$ then $r_{e_1,s+1} < r_{e_2,s+1}$. For all $e'' > e$, declare that $R_{e''}$ is not waiting for convergence at any length at stage $s + 1$.

Verification

Before we demonstrate the satisfaction of requirements R_e, Q_e, we need to show that the partial martingale m is well-defined, and for all $n, s \in \omega$,

$$\text{if} \quad m(A_s \upharpoonright n)\downarrow \quad \text{then} \quad m(A_s \upharpoonright n) \geqslant 1. \tag{2.2}$$

We clarify that in this statement, m denotes the state of the partial martingale at stage s. We prove this statement by induction on the stages s. We first claim that if R_e requires attention in Case (ii) at stage s, then m has not been defined on any string extending $A_s \upharpoonright r_{e,s}\hat{\ }0$. Suppose R_e requires attention in Case (ii) at stage s. Let $s^* - 1 < s$ be the last stage at which R_e was initialised. We choose r_{e,s^*} to be some fresh large number. In particular, m has not been defined on any string extending $A_{s^*} \upharpoonright r_{e,s^*}\hat{\ }0$. Note that $r_{e,s} = r_{e,s^*}$ and $A_{s^*} \upharpoonright r_{e,s} = A_s \upharpoonright r_{e,s}$. If R_k for $k > e$ acts at stage $t > s^*$ it may define m on $A_t \upharpoonright r_{k,t}\hat{\ }0$, but as $r_{k,t} > r_{e,t} \geqslant r_{e,s}$, it cannot define m on any string extending $A_{s^*} \upharpoonright r_{e,s}\hat{\ }0$. No R_i for $i < e$ may act between stages s^* and s as this contradicts the choice of s^*. Therefore m has not been defined on any string extending $A_s \upharpoonright r_{e,s}\hat{\ }0$ at stage s.

By the definition of A_0 and m at stage 0, we have $m(A_0 \upharpoonright n) \geqslant 2$ for all n. Furthermore, m is at least 1 on the sibling of any initial segment of A_0 (recall the definition of the *sibling* of a string, just after Definition 2.1.1). Suppose that $m(A_s \upharpoonright n) \geqslant 1$ for all n and m is at least 1 for any sibling of an initial segment of A_s. If we act for Q_e at stage s, then it is easy to see that $m(A_{s+1} \upharpoonright n) \geqslant 1$ for all n. So suppose that we act for R_e at stage s. If we act in Case 2 for R_e at stage s, then we will choose a string τ extending $A_s \upharpoonright r_{e,s}\hat{\ }0$ of some length h. By assumption, $m(A_s \upharpoonright r_{e,s}\hat{\ }0) \geqslant 1$. We then let A_{s+1} extend τ and define m such that $m(\tau) - m(A_s \upharpoonright r_{e,s}\hat{\ }0) > n_e(A_s \upharpoonright r_{e,s}\hat{\ }0)$. Then $m(A_{s+1} \upharpoonright n) \geqslant 1$ for all n.

Now suppose that we act for R_e in Subcase 1b through length l at stage s. Our martingale m wagers at most \$1 at a time, and so loses at most \$1 at a time. We decrease n_e by at least \$1 while decreasing m by at most \$1. As $m(A_s \upharpoonright l) > n_e(A_s \upharpoonright (l-1))$, we may reduce n_e's capital to \$0 while m has capital remaining. Now requirements of stronger priority than R_e may start to act. Suppose that $R_{e'}$ with $e' < e$ requires attention. If $R_{e'}$ requires attention in Case (ii) then we will act as in the previous paragraph and m will still have capital left. Otherwise, $R_{e'}$ may act in Subcase 1b at stage t after having acted in Case 2 *before* stage s. However, in this case, we would have increased m's capital by $n_{e'}(A_s \upharpoonright r_{e',s}{}^\frown 0)$ previously. Therefore, after having reduced n_e's capital to 0, we may then reduce $n_{e'}$'s capital to 0 as well, while ensuring that m still has capital remaining. This concludes the induction on the stages and the proof of (2.2).

Note that we have not yet shown that the approximation (A_s) converges to a set A. This is a consequence of the use of restraints in the construction, and the following lemma which says that each requirement R_e receives attention only finitely often.

Lemma 2.2.1. *For all $e \in \omega$, R_e receives attention only finitely often, and is met.*

Proof. Suppose by induction that $s^* - 1$ is the last stage at which R_e is injured (i.e. the least stage after which no requirement of stronger priority than R_e receives attention). If R_e never requires attention at some later stage in Case (i), then either n_e is not total, or $n_e(A) \leqslant n_e(A_{s^*} \upharpoonright r_{e,s^*})$. In either case R_e is met. Therefore suppose that R_e requires attention through l at some stage $s' \geqslant s^*$. We will act in Subcase 1a and declare R_e to be waiting for convergence at length $n_e((A_{s'} \upharpoonright r_{e,s'})^\frown 0) + 1 + (r_{e,s'} + 1) =: h_0$. As no requirement $R_{e'}$ for $e' < e$ receives attention after stage s^*, R_e will be waiting for convergence at length h_0 until, if ever, R_e requires attention in Case (ii). If R_e never requires attention after stage s' then n_e is not total. So suppose that R_e requires attention in Case (ii) at stage t'. We choose a string τ of length h_0 above $A_{t'} \upharpoonright r_e{}^\frown 0$ such that $n_e(\tau) \leqslant n_e(A_{t'} \upharpoonright r_e{}^\frown 0)$. Since n_e is a (partial) martingale, $n_e(\sigma) \leqslant n_e(A_{t'} \upharpoonright r_e{}^\frown 0)$ for at least one string σ of length h_0 above $A_{t'} \upharpoonright r_e{}^\frown 0$. Therefore such a string must exist.

We set $A_{t'+1} = \tau\,\hat{}\,1^\omega$ and define m so that $m(\tau) > n_e(\tau)$. If R_e receives attention after stage t' then it must do so in Subcase 1b. Our martingale m wagers at most \$1 at a time, and so loses at most \$1 at a time. If R_e requires attention through some $l > h_0$ at a stage $t'' > t'$ then we will again act in Subcase 1b and force n_e to lose at least \$1 while m loses at most \$1. This can happen at most $n_e(\tau)$ many times before n_e loses all its capital and can no longer bet. Thus the induction can continue, and R_e is met. □

It remains to show that m succeeds on A. For this, it suffices to show that all requirements Q_e, $e \geqslant 2$ are met.

Lemma 2.2.2. *For all $e \geqslant 2$, Q_e receives attention only finitely often, and is met.*

Proof. Suppose by induction that s^* is the least stage after which no requirement of stronger priority than Q_e receives attention. As no requirement of stronger priority than Q_e receives attention after stage s^*, the restraint r_{e,s^*} will never again be increased unless Q_e acts. If Q_e requires attention at stage $s > s^*$ then we must have $m(A_s \upharpoonright r_{e,s}) < e$. As no requirement of stronger priority receives attention after stage s^*, we must have that Q_{e-1} is satisfied, and so $m(A_s \upharpoonright r_{e,s}) = e - 1$. We have defined m to wager \$1 on all initial segments of A_s and so there is τ such that $m(\tau) = e$. We let $A_{s+1} = \tau\,\hat{}\,1^\omega$ and increase the restraints $r_{e',s+1}$ for all $e' \geqslant e$. We then have that $\tau \prec A$ and Q_e is satisfied. □

2.2.3 Integer-valued randoms not computing partial integer-valued randoms

The construction of Section 2.2.2, non-trivial as it is, admits some modifications. For example, it is not hard to add the requirement that A is 1-generic and still successfully perform the argument. This requirement can be canonically split into an infinite sequence of conditions, with corresponding strategies in the constructions which will occasionally change the approximation to A. Since the 1-genericity sub-requirements are finitary, their effect will be similar to (in fact, more benign than) the R_e requirements.

There is a 1-generic IVR which is not partial IVR.

Note that 1-generics are generalized low, so since A is Δ^0_2, it follows that

There is a low IVR which is not partial IVR.

The next modification of the construction of Section 2.2.2 results in the proof of Theorem 2.1.4 and requires more explanation. We can replace the requirements Q_e with

$Q^*_{e,k}$: If Φ^A_e is total and non-computable then m wins (at least) \$k on Φ^A_e.

Note that now m will bet on Φ^A_e rather than A. For this reason, the family of requirements $Q^*_{e,k}$ need to act under the hypothesis that Φ^A_e is total. This means that we need to implement the argument of Section 2.2.2 on a tree, where the family of requirements $Q^*_{e,k}$ lies below a 'mother-node' Q^*_e which has two outcomes, a Π^0_2 outcome i

and a Σ_2^0 outcome f. The outcome i corresponds to the fact that Φ_e^A has infinitely many expansionary stages (i.e. stages where the least n such that $\Phi_e^A(n)$ is undefined is larger than every before) while outcome f corresponds to the negation of this statement. Moreover the construction guarantees that if i is a true outcome, then Φ_e^A is total. Requirements $Q_{e,k}^*$ act as the Q_k of Section 2.2.2 while R_e are the same in the two constructions. Moreover these two requirements have a single outcome in the tree argument. The crucial point here is that if Φ_e^A is total and non-computable then Φ_e will have splitting along A, i.e. for each prefix τ of A there will be two finite extensions τ_i of τ and an argument x such that $\Phi_e^{\tau_0}(x) \neq \Phi_e^{\tau_1}(x)$. This means that before strategy $Q_{e,k}^*$ starts operating, it can secure a splitting which it can use to move away from versions of Φ_e^A on which m has not bet appropriately. In the construction of Section 2.2.2 this happened automatically as m bet on the real itself, and not its image under a Turing functional. Other than these points, the construction and verification are entirely similar to those of Section 2.2.2. Since there is no novelty in this extension of the argument of Section 2.2.2 (given the standard machinery for tree arguments and the above remarks) we leave the remaining details to the motivated reader.

2.3 Computable enumerability and IVRs

Nies, Stephan and Terwijn [38] showed that a c.e. degree is high if and only if it contains a computably random c.e. real. Moreover an analogous statement holds for partial computably random c.e. reals. In Section 2.3.1 we show that the same is true for integer-valued random c.e. reals. In other words, we give the proof of the first part of Theorem 2.1.6 that we discussed in the introduction. The proof of the remaining part of Theorem 2.1.6 (regarding partial integer-valued randoms) is deferred to Section 2.4.3, since the required machinery is similar to the one we use for the jump inversion theorems. In Section 2.3.2 we give the proof of Theorem 2.1.7. Note that by Corollary 2.1.5, for this proof it suffices to show that array computable c.e. degrees are not integer-valued random.

2.3.1 Degrees of left-c.e. integer-valued randoms

In this section we prove the part of Theorem 2.1.6, i.e. that the high c.e. degrees are exactly those c.e. degrees which contain integer-valued random left-c.e. reals. (We prove the rest of the theorem in Section 2.4.3.) The 'if' direction is a consequence of [38]. For the 'only if' direction it suffices to show that every integer-valued random left-c.e. real has high degree. Let α be an integer-valued random left-c.e. real, and $\langle \alpha_s \rangle_{s<\omega}$ a computable increasing sequence of rationals converging to α. We know that α has infinitely many 1s as it is integer-valued random, and so by speeding up the enumeration we may ensure that α_s has at least s 1's. Let $\mathrm{Tot} = \{e \mid \varphi_e \text{ is total}\}$ be the canonical Π^0_2 complete set. We build a Turing functional Γ such that for all e, $\lim_k \Gamma^\alpha(e,k) = \mathrm{Tot}(e)$. Then $\varnothing'' \leqslant_T \alpha'$ and so α is high. We also construct for each $e \in \omega$ a computable integer-valued martingale M_e. Let

$$\begin{aligned} d_{e,s} = d_e[s] &= \max\{k \mid (\forall \sigma \in 2^k)(M_e(\sigma)[s] \downarrow)\} \\ l_{e,s} &= \max\{k \mid (\forall j < k)(\varphi_e(j)[s] \downarrow)\}. \end{aligned}$$

We proceed in stages s, each consisting of two steps.

Construction at stage $s+1$ For each $\langle e,k\rangle \leqslant s$ do the following

(a) If $\Gamma^\alpha(e,k)[s] \uparrow$, define it as follows. If $l_{e,s+1} \geqslant k$ let $\Gamma^\alpha(e,k)[s+1] = 1$ with use $\gamma(e,k)[s+1] = 0$; otherwise let $\Gamma^\alpha(e,k)[s+1] = 0$ with use the maximum of $\gamma(e,k)[s]$, $\gamma(e,k-1)[s+1]$, and h, where h is the position of the first 1 of α_s after $\alpha_s \upharpoonright d_{e,s}$.

(b) If $l_{e,s+1} > l_{e,s}$, Define M_e to wager 1 dollar on $(\alpha_s \upharpoonright h)\hat{}\,1$, and bet neutrally on all other strings with length in $(d_{e,s}, h+1]$, where h is the position of the first 1 of α_s after $\alpha_s \upharpoonright d_{e,s}$.

Verification Since α is a left-c.e. real, it follows from the construction that Γ is well defined, i.e. it is consistent. We show that Γ^α is total by showing that $\lim_s \gamma(e,k)[s]$ exists for all pairs (e,k), and that $\lim_k \Gamma^\alpha(e,k) = \mathrm{Tot}(e)$. We say that a stage t is *e-expansionary* if $l_{e,t} > l_{e,t-1}$.

First suppose that $e \notin \mathrm{Tot}$. Then $\lim_s l_{e,s}$ and $\lim_s d_{e,s}$ both exist. Let $\lim_s l_{e,s} = l$ and $\lim_s d_{e,s} = d$, and suppose these limits are reached by stage s_0. Let $s_1 \geqslant s_0$ be the least stage where $\alpha_{s_1} \upharpoonright d = \alpha \upharpoonright d$. Then for all k and all stages s where $\langle e, k\rangle \geqslant s \geqslant s_1$, $\Gamma^\alpha(e,k)[s]$ is set to 0 and $\gamma(e,k)[s]$ is set to be the position of the first 1 after $\alpha_s \upharpoonright d$. As α is left-c.e., the position of the first 1 of α_s after $\alpha_s \upharpoonright d$ at any stage $s \geqslant s_1$ is at most the position of the first 1 of α_{s_1} after $\alpha_{s_1} \upharpoonright d$. Therefore $\lim_s \gamma(e,k)[s]$ exists. For all k such that $\langle e,k\rangle < s_1$, $\lim_s \gamma(e,k)[s]$ is at most $\max_{s<s_1} h_{e,s}$ where $h_{e,s}$ is the position of the first 1 of α_s after $\alpha_s \upharpoonright d_{e,s}$.

Now suppose that $e \in \mathrm{Tot}$. Then there is a sequence of stages $\langle s_i\rangle$ and a sequence $\langle h_i\rangle$ such that we define M_e to wager 1 dollar on $(\alpha_{s_i} \upharpoonright h_i)\hat{\ }1$ at stage s_i. Note that s_i is least such that $l_{e,s_i} = i$. The real α is left-c.e., so $\alpha_s \upharpoonright (h_i+1)$ can only move lexicographically to the left as s increases. Moreover, the approximation to α will never extend $(\alpha_{s_i} \upharpoonright h_i)\hat{\ }0$, and so M_e cannot lose capital along α. As α is integer-valued random, M_e does not succeed on α. If $\alpha \upharpoonright (h_i+1) = \alpha_{s_i} \upharpoonright (h_i+1)$ then M_e increases in capital by 1 dollar. Therefore there are only finitely many h_i for which $\alpha \upharpoonright (h_i+1) = \alpha_{s_i} \upharpoonright (h_i+1)$. Let i_0 be least such that $\alpha \upharpoonright (h_j+1) \neq \alpha_{s_j} \upharpoonright (h_j+1)$ for all $j \geqslant i_0$. We show that $\Gamma^\alpha(e,k) = 1$ for all $k \geqslant i_0$, thus concluding the proof.

Suppose by induction that $\Gamma^\alpha(e,i) = 1$ for all $i_0 \leqslant i < k$. At stage s_{k-1} we have $l_{e,s_{k-1}} = k-1$. For any stage t with $s_{k-1} \leqslant t < s_k$, if $\Gamma^\alpha(e,k)$ becomes undefined we set $\Gamma^\alpha(e,k)[t] = 0$ and set $\gamma(e,k)[t]$ to be the maximum of $\gamma(e,k)[t-1]$, $\gamma(e,k-1)[t]$, and the position of the first 1 of α_t after $\alpha_t \upharpoonright d_{e,t}$. Let h be the position of the first 1 in α_{s_k} after $\alpha_{s_k} \upharpoonright d_{e,s_{k-1}}$. At stage s_k we see $l_{e,s_k} = k$ and define M_e to wager

1 dollar on $(\alpha_{t'} \upharpoonright h)\hat{}1$. If α changes below $\gamma(e,k)[s_k - 1]$ at stage s_k then $\gamma(e,k)[s_k]$ will be set to at least h. Otherwise, $\gamma(e,k)[s_k - 1] \geqslant h$. Then as $k \geqslant i_0$, α changes below h at some stage $t' > s_k$. At stage t', $\Gamma^{\alpha}(e,k)$ will become undefined. At the next e-expansionary stage we set $\Gamma^{\alpha}(e,k) = 1$ with use 0. This concludes the verification and the proof of part of Theorem 2.1.6.

2.3.2 Array computable c.e. degrees do not compute integer-valued randoms

A natural class of c.e. degrees that do not contain integer-valued randoms is the class of array computable degrees. In this section we sketch the proof of this fact, which along with Corollary 2.1.5 gives Theorem 2.1.7 that was presented in the introduction.

By [25, 44] (also see [18, Proposition 2.23.12]) if A is array computable and c.e., h is a nondecreasing unbounded function and $f \leqslant_T A$, then there exists a computable approximation $(f[s])$ of f such that

$$|\{s \mid f(x)[s] \neq f(n)[s+1]\}| \leqslant h(x) \text{ for all } x. \qquad (2.3)$$

Hence given an integer-valued random B and a c.e. set A such that $B \leqslant_T A$, it suffices to define an order function h and a function $f \leqslant_T A$ such that any computable approximation $(f[s])$ to f does not satisfy (2.3). Let B be integer-valued random and suppose A is c.e. and $\Gamma^A = B$. We assume that at stage s, Γ has computed s many bits of $\Gamma^A[s]$. We define an order function h and a Turing functional

Δ such that the function $f = \Delta^A$ does not satisfy (2.3) for any computable approximation $(f[s])$ of it. Let $\langle \psi_e \rangle$ be an effective list of all binary partial computable functions. We meet the requirements

R_e**:** $(\exists x)(f(x) \neq \lim_s \psi_e(x,s) \vee |\{s \mid \psi_e(x,s) \neq \psi_e(x,s+1)\}| \geqslant h(x))$.

We define for each $e \in \omega$ an integer-valued martingale m_e. First, let us describe the strategy for R_0. We will have $h(0) = 1$. At stage 1 we define $f(0) = 1$ with use $\delta_1(0) = \gamma_1(0)$. We wait until a stage s where we see $\psi_0(0,s) = f(0)[s] = 1$. If this happens, we will want to define m_0 to put pressure on A to change so that we may redefine $f(0)$. We define m_0 to start with \$1 in capital and wager \$1 on $\Gamma^A[s] \upharpoonright 1$. If $\Gamma^A \upharpoonright 1$ changes then we get a change in A below $\gamma_1(0)$, and so a change in A below $\delta_1(0)$. We may therefore redefine $f(0)$ and so meet R_0. We assume that there will be no change in $\Gamma^A \upharpoonright 1$, and so we immediately look to see whether we can start attacking R_0 again by trying to redefine $f(1)$. At stage 2 we define $f(1) = 2$ with use $\delta_2(1) = \gamma_2(1)$. If we see no change in $\Gamma^A \upharpoonright 1$, then the martingale m_0 has \$2 on $\Gamma^A \upharpoonright 1$. We will have $h(1) = 1$. We wait until a stage s' where we see $\psi_0(0,s') = f(0)[s']$ and $\psi_0(1,s') = f(1)[s']$. If this happens, we define m_0 to wager \$1 on $\Gamma^A[s'] \upharpoonright 2$. If A changes below $\gamma_{s'}(1)$ then we may redefine $f(1)$ and meet R_0.

We would like m_0 to be total. Therefore whenever we let m_0 wager some of its capital on a string σ, we extend m_0 by letting it bet neutrally on all other strings of length at most $|\sigma|$. Now suppose that none of our previous attempts to redefine $f(0), \dots, f(x-1)$ have been successful. We wait until a stage s where we have $\psi_e(y,s) = f(y)[s]$ for all $y \leqslant x$. The use $\delta_s(x)$ will be equal to $\gamma_s(l)$ for some l. Suppose we have defined m_0 up to strings of length $l-1$ and that m_0 has \$$k$ on $\Gamma^A \upharpoonright (l-1)$. Suppose $h(x) = n$. Then we require n changes in A to redefine $f(x)$ as many times as we would like. If we let m_0 wager \$1 on $\Gamma^A \upharpoonright l$ and see A change below $\gamma_s(l)$, we can redefine $f(x)$ once. Suppose that we see this change at stage t. We lift $\delta_t(x) = \gamma_t(l+1)$. The martingale m_0 has been defined up to strings of length l, and we have $m_0(\Gamma^A_t \upharpoonright l) = k-1$. We again wait until a stage t' where $\psi_e(y,t') = f(y)[t']$ for all $y \leqslant x$. If this occurs, we now define m_0 to wager \$2 on $\Gamma^A_{t'} \upharpoonright (l+1)$. We do this so that if we do not see a change in this instance, m_0's capital becomes \$$k+1$. When we set $\delta_t(x) = \gamma_t(l+1)$ this caused $f(x')$ to become undefined for all $x' > x$. At stage $t'+1$ we define $f(x+1) = x+2$ with use $\delta_{t'+1}(x+1) = \gamma_{t'+1}(l+2)$. Therefore, if necessary we may start attacking R_0 by trying to redefine $f(x+1)$. If every time we see a change for $f(x)$ we increase our wager by \$1, after $n-1$ many changes we are left with \$$k-(1+2+\ldots+n-1) = \$k - \frac{1}{2}(n-1)n$. In attempting to get the nth change, we wager all remaining capital and require that if we do not see another change, then we end up with more than \$$k$. So we want $2(k - \frac{1}{2}(n-1)n) > k$. That is, $k > (n-1)n$. We therefore set $h(0) = 1$ and let $h(n)$ be the greatest m such that $(m-1)m < h(n-1)+1$. If we define m_0 as above then either we see all required changes, or m_0's capital increases to at least $k+1$. As B is integer-valued random, we eventually do see all changes to redefine some $f(x)$, and satisfy R_0.

Multiple requirements and interactions

In order to to deal with multiple requirements, we proceed as follows. The function $f = \Delta^A$ is a global object which must be defined on all inputs. As in the strategy above, the values $f(x)$ are changed by the action of the requirements. Suppose we satisfy R_0 by redefining $f(0)$ once. We could attempt to satisfy R_1 by further redefining $f(0)$, but at some point we must stop. We choose a fresh large number x_1, and have the strategy for R_1 try to redefine $f(x_1)$ as many times as necessary. As we saw above, the strategy for R_1 may at any one time be wanting to redefine $f(y)$ for possibly many y. We formalise this by associating to each requirement R_e at stage s an interval $I_{e,s}$ of natural numbers, so that R_e at stage s is wanting to redefine $f(x)$ for $x \in I_{e,s}$. When we are successful in redefining $x \in I_{e,s}$, we remove all $y > x$ from $I_{e,s}$. If we have not already satisfied R_e at some later stage s' and we see $\psi_e(z, s') = f(z)[s']$ for all $z \leqslant x+1$, then we add $x+1$ to $I_{e,s'}$ and attempt to redefine $f(x+1)$ as well.

Consider the requirements R_e and $R_{e'}$, with R_e of stronger priority than $R_{e'}$. We are defining martingales m_e for R_e and $m_{e'}$ for $R_{e'}$. It is possible that when Γ^A moves and we redefine some $f(k)$ for the sake of R_e that the martingale $m_{e'}$ also loses capital, even though we do not redefine some $f(j)$ for the sake of $R_{e'}$. We will therefore want to start a new version of $m_{e'}$ every time a requirement of stronger priority than $R_{e'}$ acts. We say that R_e requires attention at stage s if one of the following holds:

1. $I_{e,s} = \emptyset$.
2. for all $x \leqslant \max I_{e,s}$ we have $\psi_e(x, s) = f(x)[s]$ and
$$|\{t < s : f(x)[t] \neq f(x)[t + 1]\}| < h(x),$$
and $A_s(z) \neq A_{s-1}(z)$ for some $z \in (\delta_s(\min I_{e,s} - 1), \delta_s(\max I_{e,s})]$.
3. for all $x \leqslant \max I_{e,s}$ we have $\psi_e(x, s) = f(x)[s]$ and
$$|\{t < s : f(x)[t] \neq f(x)[t + 1]\}| < h(x),$$
and $\psi_e(\max I_{e,s} + 1, s) = f(\max I_{e,s} + 1)[s]$.

We are ready to produce the construction.

Construction

At stage 0, define $m_e(\lambda) = 1$ for all $e \in \omega$. Let $f(x)[0] = \Delta^A(x)[0] = 1$ with use $\delta_0(x) = x$ for all $x \in \omega$. Let $I_{e,1} = \emptyset$ for all $e \in \omega$. Each stage of the construction after stage 0 consists of three steps. At stage s, $s \geqslant 1$ proceed as follows:

Step 1: For all $e \leqslant s$, if a requirement of stronger priority than R_e has acted since R_e last acted, we start a new version of m_e, and define $m_e(\lambda) = 1$. Otherwise, we continue with the previous version of m_e. Let $d_{e,s}$ denote the length of the longest string for which the current version of m_e is defined.

Step 2: Let x be least such that $f(x)$ is undefined at the beginning of stage s. (If there is no such x, proceed to the next step.) Let $l = \max_{e \leqslant s} d_{e,s}$. Define $f(x)[s] = s + 1$ with use $\delta_s(x) = \gamma_s(l + 1)$.

Step 3: Let R_e be the requirement of strongest priority which requires attention at stage s. Choose the first case by which R_e requires attention.

If case 1 holds, choose a fresh large number x_e and let $I_{e,s+1} = \{x_e\}$.

If case 2 holds, then let $x \in I_{e,s}$ be least such that $A_s(z) \neq A_{s-1}(z)$ for some $z \in (\delta_s(x-1), \delta_s(x)]$. Let $f(x) = s+1$ with use $\delta_{s+1}(x) = \gamma_{s+1}(d_{e,s} + 1)$. We have that

$$m_e(\Gamma^A[s] \upharpoonright d_{e,s}) < m_e(\Gamma^A[s] \upharpoonright (d_{e,s} - 1)).$$

If $m_e(\Gamma^A[s] \upharpoonright d_{e,s}) \neq 0$, let $n_e = \max_{i \leqslant d_{e,s}} h(m_e(\Gamma^A[s] \upharpoonright i))$. Suppose that j is such that $n_e = h(m_e(\Gamma^A[s] \upharpoonright j)$ and let $n'_e = |\{m_e(\Gamma^A[s] \upharpoonright i) : j \leqslant i \leqslant d_{e,s}\}|$. Then we have received n'_e of the n_e permissions required to redefine $f(x)$ at least $h(x)$ many times. If $n'_e = n_e - 1$, then define m_e to wager $\Gamma^A[s] \upharpoonright d_{e,s}$ dollars on $\Gamma^A[s] \upharpoonright (d_{e,s}+1)$. Otherwise let $w = m_e(\Gamma^A[s] \upharpoonright (d_{e,s}-1)) - m_e(\Gamma^A[s] \upharpoonright d_{e,s})$ and define m_e to wager $\$(w + 1)$ on $\Gamma^A[s] \upharpoonright (d_{e,s} + 1)$. If $m_e(\Gamma^A[s] \upharpoonright d_{e,s}) = 0$, let m_e bet neutrally on all other strings of length $d_{e,s} + 1$. Let $I_{e,s+1} = [\min I_{e,s}, x]$.

If case 3 holds, then let $I_{e,s+1} = I_{e,s} \cup \{\max I_{e,s} + 1\}$. Define m_e to wager \$1 on $\Gamma^A[s] \upharpoonright (d_{e,s} + 1)$.

In any case, let $I_{e',s+1} = \varnothing$ for all $e' > e$.

Verification

We need to show that for all $e \in \omega$, R_e is satisfied. Assume by induction that stage s^* is the last stage at which a requirement of stronger priority than R_e acts. Assume for all $s \geqslant s^*$ that $\psi_e(x, s) = f(x, s)$ for all $x \leqslant \max I_{e,s}$. At stage $s^* + 1$ we will define a new version of m_e, which will be the final version. At every stage after $s^* + 1$, we define more of m_e. Therefore m_e is total. As Γ^A is integer-valued random, $m_e(\Gamma^A) = \sup\{m_e(\Gamma^A \upharpoonright i) : i \in \omega\} < \infty$. Let $\sup\{m_e(\Gamma^A \upharpoonright i) : i \in \omega\} = k$ and i_0 be such that $m_e(\Gamma^A \upharpoonright i_0) = k$. Suppose s_0 is least such that $s_0 \geqslant s^* + 1$ and $\Gamma^A[s_0] \upharpoonright i_0 = \Gamma^A \upharpoonright i_0$, and x is such that $\delta_{s_0}(x) = \gamma_{s_0}(d_{e,s_0} + 1)$. Then $f(x)$ is redefined $h(x)$ many times and R_e is satisfied.

2.4 Jump inversion for integer-valued randoms

Jump inversion for Martin-Löf randoms was discovered in [30, 21] and was generalized in [4]. Every degree which is c.e. in and above $0'$ contains the jump of some Martin-Löf random Δ^0_2 set. Hence the same holds for the integer-valued randoms. However in this case we can obtain a stronger jump inversion theorem by requiring that the 'inverted' degrees are c.e. Note that this stronger theorem does not hold for Martin-Löf randoms since $0'$ is the only c.e. degree containing a Martin-Löf random. Moreover it does not hold for computable randomness or Schnorr randomness, since by [38] the only c.e. degrees that contain such randoms are high. integer-valued randomness is the strongest known randomness notion for which jump inversion holds with c.e. degrees.

Since the argument is somewhat involved, we present it in two steps. In Section 2.4.1 we discuss the strategy for controlling the jump of an integer-valued random of c.e. degree. This argument gives a low c.e. degree which contains an integer-valued random. It is a finite injury construction, and the hardest of the two steps. Our argument actually shows the stronger result that there is a low c.e. weak truth table degree which contains an integer-valued random. We can then add coding requirements in order to prove the full jump inversion theorem, which we present in full detail in Section 2.4.2. This construction is a tree argument which uses the strategies of Section 2.4.1 for ensuring that the jump of the constructed set is below the given Σ^0_2 set, combined with standard coding requirements which deal with the remaining requirements.

2.4.1 A low c.e. degree containing an integer-valued random

We build an integer-valued random A of low c.e. degree. In fact, we build an integer-valued random A and a c.e. set B such that $A \equiv_{wtt} B$. Let $\langle m_e \rangle$ be an effective list of all partial integer-valued martingales. In order to ensure that A is integer-valued random it suffices to satisfy the following requirements:

R_e**:** if m_e is total, then m_e does not succeed on A;

N_e**:** $(\exists^\infty s)\ (\Phi^A_e(e)[s]\downarrow) \implies \Phi^A_e(e)\downarrow.$

We order the requirements as $R_0 > N_0 > R_1 > N_1 > \dots$ and begin by setting $A_1 = 1^\omega$. To meet R_0, we observe the values of the martingale m_0. If m_0 increases its capital along A, we change A to force m_0 to lose capital. As m_0 is integer-valued, if it loses capital, it must lose at least \$1. Thus if we can force m_0 to lose capital every time we act, we need only act for R_0 finitely many times. As we are building reductions Γ and Δ such that $\Gamma^B = A$ and $\Delta^A = B$, to change A we will need to change B. Once we have changed B, we will then need to change A again to record this fact. To satisfy the requirement N_e we use the usual strategy of preserving the restraint $\varphi_e^A(e)[s]$ at all but finitely many stages. As the strategy for an R-requirement is finitary, this can be done easily.

The finite injury construction

In order to help with the definition of the reductions, we make use of *levels* $\langle l_i \rangle_{i<\omega}$ and $\langle d_i \rangle_{i<\omega}$. We calculate the size of the levels below. We set $\gamma(l_i) = d_i$ and $\delta(d_i) = l_{i+1}$. We say that we act for requirement R_e *at level* l_{i+1} at stage s if we change A to decrease m_e's capital from $A \upharpoonright l_i$ to $A \upharpoonright l_{i+1}$. That is, $m_e(A_s \upharpoonright l_{i+1}) > m_e(A_s \upharpoonright l_i)$ and $m_e(A_{s+1} \upharpoonright l_{i+1}) < m_e(A_{s+1} \upharpoonright l_i)$. We act at level l_{i+1} only for the sake of the requirements $R_0, \dots, R_i$. Once we have acted at level l_{i+1}, we enumerate an element from $[d_i, d_{i+1})$ into B. To record this change in B, we let A extend a string of length l_{i+2} which has not yet been visited. So that the reduction Δ is consistent, we must not let A extend a string which is *forbidden*, that is, a string σ such that

$\Delta^\sigma \not\prec B_{s+1}$. We carefully define the levels l_i so that the action from the requirements never forces us to extend a forbidden string.

Before we define $\langle l_i \rangle_{i<\omega}$ and $\langle d_i \rangle_{i<\omega}$, we lay out the construction (in terms of these unspecified parameters and a function d defined below). Later in this section we discuss the various properties that these parameters need to satisfy in order for the construction to be successful (i.e. produces sets A, B which satisfy the requirements R_e, N_e).

We have for every $e \in \omega$ and every stage s a restraint $r_{e,s}$. We say that R_e requires attention at level l_{i+1} at stage s if

1. $m_e(\sigma)[s] \downarrow$ for all strings σ of length $\leqslant l_{i+1}$,
2. $l_{i+1} \geqslant r_{e,s}$
3. $m_e(A_s \upharpoonright l_{i+1}) > m_e(A_s \upharpoonright l_i)$.

We say that R_e requires attention at stage s if it does so at some level. We say that N_e requires attention at stage s if $\Phi_e^A(e)[s] \downarrow$. Recall the definition of the *sibling* of a string, just after Definition 2.1.1.

Construction Let $\gamma(l_i) = d_i$ and $\delta(d_i) = l_{i+1}$. At stage 0, let $A_1 = 1^\omega$ and $r_{e,1} = l_e$ for all e.

Stage s, $s \geqslant 1$: Find the requirement of strongest priority which requires attention at stage s. (If there is no such requirement, proceed to the next stage.)

Case 1: If this is R_e, let l_{i+1} be least such that R_e requires attention at level l_{i+1} at stage s. Let $l \in (l_i, l_{i+1}]$ be least such that $m_e(A_s \upharpoonright l) > m_e(A_s \upharpoonright l_i)$. Choose a string τ of length l_{i+1} which extends the sibling of $A_s \upharpoonright (l-1)$ such that the minimum of all $d(\tau, \mu)$, where μ is any forbidden string of length l_{i+1} extending $A_s \upharpoonright l_i$, is as large as possible. Enumerate an element of $[d_i, d_{i+1})$ into B. Choose a string ρ of length l_{i+2} extending τ such that $\rho \not\prec A_t$ for all $t < s$, and the minimum of all $d(\rho, \mu)$, where the minimum is taken over forbidden strings μ of length l_{i+2} extending τ, is as large as possible. Set $A_{s+1} = \rho 1^\omega$. For all $e' \geqslant e$ with $r_{e',s} \leqslant l_{i+1}$, let $r_{e',s+1} = l_{i+1}$.

Case 2: If this is N_e, for all $e' > e$ with $r_{e',s} \leqslant \varphi_e^A(e)[s]$, let $r_{e',s+1} = \varphi_e^A(e)[s]$.

In the following section we give the remaining specifications and analysis of the construction, as well as the verification.

The calculation of the levels l_i and d_i for a successful construction

In the following we calculate the levels l_i, d_i, and depict this process in Figure 2.2. Suppose that we act at level l_{i+1} for R_e and naively let A extend a string τ of length l_{i+1} whose sibling is forbidden. Consider the situation where m_0 increases its capital on the very last bit of $A_s \upharpoonright l_{i+1}$, loses capital on τ, and is neutral on all other strings of length l_{i+1}. We will not be able to change A to extend τ's sibling, as this string is forbidden. However, we do not want to change A so that m_0 is neutral, as we would like the action for R_0 to be finitary. To avoid such a situation we must be more careful in how we change A.

In particular, we must ensure that A is kept in some sense "far away" from forbidden strings. This is made precise below.

We first calculate an upper bound on the number of forbidden strings of length l_{i+1} which can occur above a nonforbidden string of length l_i. Our upper bound will not be strict. A string σ of length l_{i+1} becomes forbidden if Δ^σ is no longer giving correct B-information. As $\delta(d_i) = l_{i+1}$, Δ^σ will be incorrect only if we enumerate an element into B below d_i, which occurs only when we act for a requirement at some level $\leqslant l_i$. We will act for R_e at level l_{i+1} only when we see m_e halt on all strings of length l_{i+1}, and so if A no longer changes below l_i, we will act for R_e at level l_{i+1} at most once. As we act at level l_{i+1} only for the sake of requirements $R_0, \ldots, R_i$, if A no longer changes below l_i, we can act at level l_{i+1} at most $i + 1$ times. After acting at a level l_j for some $j \leqslant i$, we allow $R_0, \ldots, R_i$ to act at level l_{i+1} again. We begin with $A_1 = 1^\omega$. Suppose we act $i + 1$ times at level l_{i+1}. We then act at level l_i. We act another $i + 1$ times at level l_{i+1} before we again act at level l_i. We can act at level l_i at most i times. This can continue until we get to level l_1, where we can change A once below l_1 for the sake of requirement R_0. Therefore we act at a level $\leqslant l_i$ $2.3.4. \ldots \cdot (i + 2) = (i + 2)!$ many times, and there are at most

$$f_{i+1} = \textstyle\sum_{j=0}^{i} (j+1)!$$

many forbidden strings of length l_{i+1}. Note that for any $k \in \omega$ we may enumerate all partial integer-valued martingales with initial capital k. We therefore may assume that our list $\langle m_e \rangle$ of all partial integer-valued martingales comes with a computable intial capital, $m_e(\lambda)$. As a martingale may at most double its capital in a single bet, the upper bound on m_e's capital at a string of length n is $2^n m_e(\lambda)$.

We now show how a martingale can force us "closer" to a forbidden string. Suppose at stage s that A_s extends the string ν of length l_{i+1}, and there is a forbidden string μ of length l_{i+2} above. For simplicity, suppose that A_s extends the leftmost string of length l_{i+2} which extends ν, and that μ is the rightmost string of length l_{i+2} which extends ν. If R_j requires attention at level l_{i+2}, we would like to choose a string τ of length l_{i+2} with $m_j(\tau) < m_j(\nu)$. The problem is the following. Suppose that m_j increases its capital on all string of length l_{i+2} which extend $\nu 0$. We recall Kolmogorov's inequality, as stated in Theorem 6.3.3 of [18].

Theorem 2.4.1 (Kolmogorov's inequality)**.** *Let d be a martingale. For any string σ and any prefix-free set S of extensions of σ, we have $\sum_{\tau \in S} 2^{-|\tau|} d(\tau) \leqslant 2^{-|\sigma|} d(\sigma)$.*

By Kolmogorov's inequality (with $\nu 0 = \sigma$ and S the set of strings of length l_{i+2} which extend $\nu 0$ in the above), this must mean that $m_j(\nu 0) > m_j(\nu)$, and so $m_j(\nu 1) < m_j(\nu)$. If m_j has sufficient capital at $\nu 1$, it may then increase its capital above $m_j(\nu)$ on all strings of length l_{i+2} which extend $\nu 10$. Again by Kolmogorov's inequality we have $m_j(\nu 10) > m_j(\nu 1)$ and so $m_j(\nu 11) < m_j(\nu 1) < m_j(\nu)$. Now m_j is an integer-valued martingale, and so after doing this finitely many times, say n times, we have $m_j(\nu 1^n) < \frac{1}{2} m_j(\nu)$ and so m_j cannot increase its capital above $m_j(\nu)$ on all strings of length l_{i+2} which extend $\nu 1^n 0$. If $l_{i+2} \geqslant l_{i+1} + n + 1$, then we can pick a string τ extending $\nu 1^n 0$ with $m_j(\tau) < m_j(\nu)$ and which is not forbidden. For two strings α and β of length l, let $d(\alpha,\beta)$ be $l - b$, where b is the length of the longest common initial segment. Then in this situation, we have $d(A_s,\mu) = l_{i+2} - l_{i+1}$ and $d(\tau,\mu) \leqslant l_{i+2} - l_{i+1} - n$. Therefore m_j has forced A distance n closer to the forbidden string μ. Now R_j might not be the only requirement which can act at level l_{i+2}. We will then need to calculate the distance that the other martingales may move A, and ensure that l_{i+2} is high enough.

We calculate a bound on how far an integer-valued martingale m may move A. If $m(\nu) = k$ and m increases its capital to $k+1$ on all strings extending $\nu 0$, then $m(\nu 1) = k - 1$. If $m(\nu 1) \geqslant 2$ then m can increase its capital to $k+1$ on all strings extending $\nu 10$. Then $m(\nu 11) = k - 1 - 2$. If $m(\nu 1) \geqslant 4$ then m can increase its capital to $k+1$ on all strings extending $\nu 110$. Then $m(\nu 111) = k - 1 - 2 - 4$. When $m(\nu 1^n) = k - 1 - 2 - \ldots - 2^{n-1} < \frac{1}{2}k$, m is not able to increase its capital to $k+1$ on all strings extending $\nu 1^n 0$. We let $n(k) = (\mu n)(k - 1 - 2 - \ldots - 2^{n-1} < \frac{1}{2}k)$. Then m can move A at most a distance $n(k)$. In the case where A_s is not the leftmost string extending ν and μ is not the rightmost string extending ν, a similar argument shows that m_e can still move A a distance of at most $n(k)$. The only difference is that m_e would then need to bet against the initial segments of μ which are of length greater than $l_{i+2} - l_{i+1} - d(A_s,\mu)$.

Suppose we act at level l_{i+1} at stage s and let A extend the string ν of length l_{i+1} which has forbidden strings of length l_{i+2} above. We enumerate an element of $[d_i, d_{i+1})$ into B. To record this change in B, we choose a string ρ of length l_{i+2} which has not been visited before, and which is as far from any forbidden string as possible. We have that there are at most f_{i+2} many forbidden strings of length l_{i+2} above a nonforbidden string of length l_{i+1}. Let $x = (\mu x)(2^x \geqslant f_{i+2})$. By the counting argument for f_{i+1} given above, if $l_{i+2} - l_{i+1} = h > x$, then there is a string of length l_{i+2} which has not been visited yet, and which is at least distance $h - x$ from a forbidden string. Now suppose that R_j requires attention at level l_{i+2}. We know that there are no forbidden strings above $\rho \restriction l_{i+1} + x + 1$, and so if we can reduce m_j by moving to a string which is still above $\rho \restriction l_{i+1} + x + 1$, we will do so. Otherwise, m_j will move us closer to a forbidden string. The bound on the capital of m_j at $A_s \restriction (l_{i+1} + x + 1)$ is $2^{l_{i+1}+x+1} m_j(\lambda)$. So we know that m_j may move us a distance at most $n(2^{l_{i+1}+x+1} m_j(\lambda))$ towards a forbidden string. If $l_{i+2} - l_{i+1} > x + n(2^{l_{i+1}+x+1} m_j(\lambda))$ then we will be able to choose a nonforbidden string ρ' which decreases m_j. Suppose that R_k, which is of stronger priority than R_j, requires attention at level l_{i+2} at stage s'. Our reasoning is similar to the previous case. Let $n_0 = n(2^{l_{i+1}+x+1} m_j(\lambda))$. We know that there are no forbidden strings above $\rho' \restriction l_{i+1} + x + n_0 + 1$, and so if we can reduce m_k by moving to a string which is still above $\rho' \restriction l_{i+1} + x + n_0 + 1$, we will do so. Otherwise, m_k will move us closer to a forbidden string. The bound on the capital of m_k at $A_{s'} \restriction (l_{i+1} +$

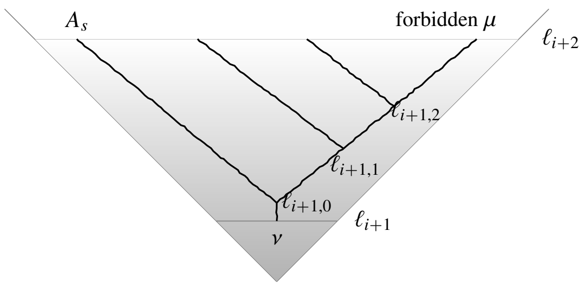

Figure 2.2: Calculating the levels l_i and avoiding the forbidden strings.

$x+n_0+1)$ is $2^{l_{i+1}+x+n_0+1}m_k(\lambda)$. So we know that m_k may move us a distance at most $n(2^{l_{i+1}+x+n_0+1}m_k(\lambda))$ towards a forbidden string. If $l_{i+2}-l_{i+1} > x+n_0+n(2^{l_{i+1}+x+n_0+1}m_k(\lambda))$ then we will be able to choose a nonforbidden string ρ'' which decreases m_k. We will need l_{i+2} to be large enough so that we can always move in this way for any requirement which might act at level l_{i+2}.

The requirements $R_0,\dots,R_i$ can act at level l_{i+1}. We do not know the order in which the requirements may act, so we will have to take the maximum of the capitals of the martingales $m_0,\dots,m_i$ in our calculation. We illustrate this definition in Figure 2.2 . We set $l_0=0$. Given l_i, we set $l_{i+1,0}=l_i+(\mu x)(2^x \geqslant f_{i+1})$ and for $0\leqslant j\leqslant i$,

$$l_{i+1,j+1}=l_{i+1,j}+\max_{k\leqslant i}\ n(2^{l_{i+1,j}+1}m_k(\lambda))$$

and let $l_{i+1}=l_{i+1,i+1}+1$. The levels d_i are chosen so that we can enumerate an element into $[d_i,d_{i+1})$ every time we act at level l_{i+1}. This calculation is the same as that of f_{i+1}. Let $d_0=0$. Given d_i, let $d_{i+1}=d_i+\sum_{j=0}^{i}(j+1)!$.

Verification of the finite injury construction

First, we show that $A \equiv_{wtt} B$. As in the calculation of f_{i+1}, we can act at level l_{i+1} at most $\sum_{j=0}^{i}(j+1)!$ many times. We have that $d_{i+1} = d_i + \sum_{j=0}^{i}(j+1)!$ for all i. Every time we act at level l_{i+1} and change A below l_{i+1}, we enumerate an element from $[d_i, d_{i+1})$ into B. The uses $\gamma(l_i)$ are clearly computable, and so we have $\Gamma^B = A$ via the weak truth-table functional Γ. For the other reduction, note that the consistency of Δ is a consequence of A never extending a forbidden string. Again the uses $\delta(d_i)$ are computable and so $\Delta^A = B$ via the weak truth-table functional Δ.

Next we argue that all N_e requirements are met. Suppose inductively that all requirements of stronger priority than N_e do not act after a certain stage t. If at some stage s after stage t the computation in requirement N_e halts, then a restraint $r_{e,s}$ will be erected so that the use of the computation is protected from further enumerations into A. Therefore in that case the computation actually halts. Therefore N_e is met, and this concludes the induction step.

It remains to show that for every $e \in \omega$, R_e is satisfied. Suppose by induction that all requirements of stronger priority than R_e do not act after stage s^*. Let i_0 be least such that $l_{i_0} \geqslant r_{e,s^*}$. We show that $m_e(A) \leqslant m_e(A_{s^*} \upharpoonright l_{i_0})$. Suppose at stage $s \geqslant s^*$ we see m_e increase its capital beyond $m_e(A_{s^*} \upharpoonright l_{i_0})$. Then R_e will require attention at stage s. Suppose that R_e requires attention at level l_{i+1} at stage s. Let $t < s$ be the last stage at which we acted for some requirement at a level below l_{i+1}. Suppose we acted at level l_{j+1}. At stage t we chose some string ρ of length l_{j+2} and let $A_t = \rho 1^\omega$. Then $A_{s-1} \upharpoonright l_i = A_t \upharpoonright l_i$. If $j = i - 1$ then we chose ρ which would have been at least distance $l_{i+1} - l_i - x$, where $x = (\mu x)(2^x \geqslant f_{i+1})$, from any forbidden string of length l_{i+1} (that is, ρ and any forbidden string have a common initial segment of length at most $l_i + x$). Otherwise $j < i - 1$ and there is no forbidden string of length l_{i+1} above $A_s \upharpoonright l_i$. Suppose that between stages t and $s - 1$ inclusive we acted at level l_{i+1} k many times. We have that $k < i + 1$. Then $A_s \upharpoonright l_i$ and any forbidden string have a common initial segment of length at most $l_{i+1,k}$. Let $l \in (l_i, l_{i+1}]$ be least such that $m_e(A_s \upharpoonright l) > m_e(A_s \upharpoonright l_i)$. If $l > l_{i+1,k} + 1$, then there is a string τ above $A_s \upharpoonright l_{i+1,k} + 1$ with $m_e(\tau) < m_e(A_s \upharpoonright l)$ and which is not forbidden. Otherwise m_e can move us at most distance $n(2^{l_{i+1,k}+1} m_e(\lambda))$ closer to a forbidden string. We have that $l_{i+1} > l_{i+1,k} + n(2^{l_{i+1,k}+1} m_e(\lambda))$, so we can find a nonforbidden string τ with $m_e(\tau) < m_e(A_s \upharpoonright l)$. Restraints are then imposed so that R_e and no other requirement of weaker priority may act at level l_{i+1} after stage s. Therefore $A \upharpoonright l_{i+1} = A_{s+1} \upharpoonright l_{i+1}$ and so $m_e(A) < m_e(A_{s^*} \upharpoonright l_{i_0})$.

2.4.2 The full jump inversion theorem for integer-valued randoms

Given a set $S \geqslant_T 0'$ which is c.e. in $0'$ we show how to construct an integer-valued random set A of c.e. degree such that $A' \equiv_T S$. Along with A, we build a c.e. set B such that $A \equiv_T B$, and show that $B' \equiv_T S$. Let $\langle m_e \rangle$ be an effective enumeration of all partial computable integer-valued martingales. So that A is integer-valued random, we meet the requirements

R_e: If m_e is total, then m_e does not succeed on A.

We also build wtt-reductions Γ and Δ such that $\Gamma^B = A$ and $\Delta^A = B$. For the requirement $S \leqslant_T B'$, we build a functional Λ and meet the requirements

P_e: $\lim_t \Lambda^B(e,t) = S(e)$.

The basic strategy for a P-requirement is as follows. As S is c.e. in and above $0'$, we know that S is Σ^0_2. Therefore there is some computable approximation $\{S_i\}_{i\in\omega}$ such that $n \in S$ if and only if there is an s such that $n \in S_t$ for all $t > s$. We define $\Lambda^B(e,s) = 1$ for larger and larger s with some large use $\lambda(e,s)$. If we see at some stage u that $e \notin S_u$ and $u > t$, then we enumerate $\lambda(e,t)$ into B and redefine $\Lambda^B(e,t) = 0$ with use -1, i.e. the axiom defining $\Lambda^B(e,t) = 0$ does not depend on B.

For the requirement $B' \leqslant_T S$ we *attempt* to meet the requirements

N_e: $(\exists^\infty s)(\Phi^B_e(e)\downarrow) \implies \Phi^B_e(e)\downarrow$.

We attempt to meet these as usual by restraining B below the use $\varphi^B_e(e)[s]$ whenever we see $\Phi^B_e(e)[s]\downarrow$. Although we will not actually meet these requirements (doing so would mean that $B' \equiv_T \varnothing'$), trying to meet the requirements will allow us to show that $B' \leqslant_T S$.

The priority tree

The construction will use a tree of strategies. To define the tree, we specify recursively the association of nodes to requirements, and specify the outcomes of nodes working for particular requirements. To specify the priority ordering of nodes, we specify the ordering between outcomes of any node. We order the requirements as

$$R_0 > P_0 > N_0 > R_1 > P_1 > N_1 > \cdots$$

and specify that all nodes of length k work for the k^{th} requirement on the list. We will have nodes dedicated to R-, P-, and N-requirements. A node dedicated to a P-requirement will have the ∞ outcome, corresponding to enumerating infinitely many markers $\lambda(e, s)$, and the f outcome, for when only finitely many markers are enumerated. Suppose that the node α works for P_e and β works for N_f with $\alpha \prec \beta$. If f is the true outcome of α and $\alpha\hat{\ } f \preccurlyeq \beta$, then only finitely many markers are enumerated, and β does not need to worry about the computations it sees being destroyed infinitely many times. Now suppose that ∞ is the true outcome of α and $\alpha\hat{\ }\infty \preccurlyeq \beta$. Then β will be guessing that P_e will enumerate all its unrestrained markers $\lambda(e, s)$ into B. It will then not believe a computation $\Phi_e^B(e)[s]$ until it sees that all unrestrained markers below the use $\varphi_e^B(e)[s]$ have been enumerated. This is formalized with the definition of a β-correct computation below. The outcomes of R- and N-nodes are $\ldots < n < \ldots < 1 < 0$, corresponding to the restraint they impose on B.

Making the sets A, B of the same degree

In Section 2.4.1 we discussed the proof that there is a low c.e. degree containing an integer-valued random set. As in that argument, we make use of *levels* in the definition of the reductions Γ and Δ. We slightly adjust the definition of the levels because we now must also enumerate the markers $\lambda(e, s)$ into B. We increase the size of each interval $[d_i, d_{i+1})$ to accommodate a coding marker. The coding markers will be chosen to be d_i for some $i \in \omega$. Now that we are also enumerating coding markers into B, we also adjust the definition of the levels l_i. Enumerating the coding markers will cause more strings to become *forbidden*. We recalculate the upper bound on the number of forbidden strings of length l_{i+1}. As before, the requirements $R_0, \ldots, R_i$ may act at level l_{i+1}. We calculated in Section 2.4.1 that we may act at most $(i + 2)!$ many times at level l_{i+1}. When the coding marker d_i is enumerated, the requirements $R_0, \ldots, R_i$ may act again at level l_{i+1}. Therefore we may act at most $2(i+2)!+1$ many times at level l_{i+1}. So there are at most $\sum_{j=0}^{i}(2(j + 1)! + 1)$ many forbidden strings of length l_{i+1}. Letting $f'_{i+1} = \sum_{j=0}^{i}(2(j+1)!+1)$, we calculate the levels l_i as before. We let $l_0 = 0$, and given l_i, we let $l_{i+1,0} = l_i + (\mu n)(2^n \geqslant f'_{i+1})$ and for $0 \leqslant j \leqslant i$ we let

$$l_{i+1,j+1} = l_{i+1,j} + \max_{k\leqslant i} n(2^{l_{i+1,j}+1} m_k(\lambda))$$

and $l_{i+1} = l_{i+1,i+1} + 1$. Set $d_0 = 0$. Given d_i, let $d_{i+1} = d_i + 2(i + 1)! + 1 + 1$.

Coordination and restraint on the tree

The action for R-requirements will otherwise be identical with the construction of Section 2.4.1. Nodes working for R-requirements will also have to be wary of coding done by P-nodes above. Suppose β, working for R_f, is below the ∞ outcome of α, which is working for P_e. If we see the martingale m_f increase its capital along A_s and wish to enumerate an element of $[d_i + 1, d_{i+1})$ into B to change A, we will wait until all unrestrained markers $\lambda(e, s)$ below d_i have been enumerated into B before changing A for the sake of R_f. If σ is accessible at stage s, we let

$$r(\sigma, s) = \max\{\sigma(|\alpha|) \mid \alpha \prec \sigma \text{ is an } N\text{-node or an } R\text{-node}\}.$$

We say that a computation $\Phi_e^B(e)[s]$ is σ-correct if for every P-node α such that $\alpha\hat{\ }\infty \preccurlyeq \sigma$, $r(\alpha, s) < \lambda(e,t) < \varphi_e^B(e)[s]$ implies $\lambda(e,t) \in B_s$. Recall the definition of the *sibling* of a string, just after Definition 2.1.1.

Construction of the sets A, B

At stage 0 we set $A_1 = 1^\omega$, $B_1 = \varnothing$, and let $r_{e,1} = l_e$ for all e. Moreover we set $\Lambda^B(0,0) = 0$ with use d_1. Each stage $s \geqslant 1$ is conducted in two steps:

Step 1: For $e, t \leqslant s$, if $\Lambda^B(e,t)$ is undefined at stage s, then let $\Lambda^B(e,t) = 1$ with some fresh large use $\lambda(e,t)$ equal to d_i for some $i \in \omega$.

Step 2: We let the collection of accessible nodes δ_s be an initial segment of the tree of strategies. Let σ be a node which is accessible at stage s. We describe the action that σ takes, and if $|\sigma| < s$, then we specify which immediate successor of σ is also accessible at stage s; otherwise, we proceed to the next stage.

Suppose first that σ works for R_e. Let k be least such that $d_k \geqslant r(\sigma, s)$. If

1. for all P-nodes α such that $\alpha\,\hat{}\,\infty \preccurlyeq \sigma$, $r(\alpha, s) < \lambda(e,t) < d_k \implies \lambda(e,t) \in B_s$, and
2. there is $l > l_{k+1}, r_{e,s}$ such that
   (a) $m_{e,s}(\sigma) \downarrow$ for all strings σ of length l, and
   (b) $m_e(A_s \upharpoonright l) > m_e(A_s \upharpoonright (l-1))$,

then let i be such that $l \in (l_i, l_{i+1}]$ (if there is more than one such l, we choose the least). Choose a string τ of length l_{i+1} which extends the sibling of $A_s \upharpoonright (l-1)$ such that the minimum of all $d(\tau, \mu)$, where μ is any forbidden string of length l_{i+1} extending $A_s \upharpoonright l_i$, is as large as possible. Enumerate an element of $[d_i + 1, d_{i+1})$ into B. Choose a string ρ of length l_{i+2} extending τ such that $\rho \not\prec A_t$ for all $t < s$, and the minimum of all $d(\rho, \mu)$, where the minimum is taken over forbidden strings μ of length l_{i+2} extending τ, is as large as possible. Set $A_{s+1} = \rho 1^\omega$. For all $e' \geqslant e$ with $r_{e',s} \leqslant l_{i+1}$, let $r_{e',s+1} = l_{i+1}$. The string $\sigma\hat{}d_{i+1}$ is accessible at stage s.

Now suppose that σ works for P_e. For all $e, t \leqslant s$, if $\lambda(e,t) > r(\sigma, s)$, $e \notin S_s$ and we have $\Lambda^B(e,t)[s] = 1$, then enumerate $\lambda(e,t)$

into B and define $\Lambda^B(e,t) = 0$ with use -1. Suppose that $\lambda(e,t) = d_i$. Choose a string ρ of length l_{i+1} extending $A_s \upharpoonright l_i$ such that $\rho \not\prec A_t$ for all $t < s$, and the minimum of all $d(\rho,\mu)$, where the minimum is taken over forbidden strings μ of length l_{i+1} extending $A_s \upharpoonright l_i$, is as large as possible. Set $A_{s+1} = \rho\hat{}1^\omega$. If a marker was enumerated, let $\sigma\hat{}\infty$ be accessible at stage s. Otherwise let $\sigma\hat{}f$ be accessible at stage s.

Finally suppose that σ works for N_e. If $\Phi_e^B(e)[s]\downarrow$ via a σ-correct computation, then let $\sigma\hat{}\varphi_e^B(e)[s]$ be accessible at stage s. Otherwise, let $\sigma\hat{}0$ be accessible at stage s.

Verification of the construction

By the construction, the set B is c.e. We verify that $A \equiv_T B$, that A is integer-valued random, and that $A' \equiv_T S$. First, we establish the existence of a 'true path'.

The leftmost path which is visited infinitely often exists. (2.4)

As there are only finitely many outcomes of a P-node, we need to verify that the restraint imposed by an N- or R-node comes to a limit. Let σ work for R_e, and suppose by induction that no node to the left of σ is visited after stage s_0, and that $\lim_s r(\sigma,s)$ exists. We must have $r(\sigma,s) = r(\sigma,s_0)$ and $r_{e,s} = r_{e,s_0}$ for all $s \geqslant s_0$. Let k be such that $d_k \geqslant r(\sigma,s_0)$. Whenever we act for R_e at level l_i for $i \geqslant k+1$, m_e's capital decreases by at least \$1 and we increase restrains for all R-requirements of weaker priority. The only elements which may be enumerated below the restraints R_e places on B are coding markers belonging to P-requirements stronger than R_e. However, if R_e is below the ∞ outcome of P_e, then R_e waits until all unrestrained markers below d_i enter B before acting at level l_{i+1}. Therefore the only markers which enter below R_e's restraint belong to those P-requirements with σ below the f outcome. By induction, we do not visit any node to the left of σ after stage s_0, and so no such strategy may act after stage s_0 and enumerate a coding marker below R_e's restraint. Therefore we act for R_e only finitely many times after stage s_0. Similarly, as we require the computations N-nodes observe to be σ-correct, if σ works for N_e and is on the true path, it will increase its restraints only finitely many times. This concludes the proof of (2.4).

Let the true path, TP, be the leftmost path visited infinitely often. The proof of (2.4) shows that we act only finitely often for any R-requirement. Therefore $m_e(A) < \infty$ and R_e is satisfied for all $e \in \omega$. Therefore

the set A is integer-valued random. (2.5)

The use of the systems of levels l_i, d_i in the construction define Turing reductions $A \leqslant_T B$ and $B \leqslant_T A$ respectively, with computable use. So that by an induction on the stages of the construction we have

the sets A and B are in the same weak truth table degree. (2.6)

It remains to show that $S \equiv_T B'$. First, we show that $S \leqslant_T B'$.

Let $\sigma \prec \mathrm{TP}$ be devoted to P_e, and suppose that no node to the left of σ is visited after stage s_0. As the restraints set by R- and N-requirements are finite, $r(\sigma, s_0)$ is finite and $r(\sigma, s) = r(\sigma, s_0)$ for all $s \geqslant s_0$. Therefore σ may enumerate all but finitely many markers if it wishes. Therefore for all $e \in \omega$, $\lim_t \Lambda^B(e,t) = S(e)$. It remains to show that $B' \leqslant_T S$. We have

$$e \in B' \Leftrightarrow \Phi_e^B(e)\downarrow \Leftrightarrow (\exists t)(\Phi_e^B(e)[t]\downarrow \wedge B \upharpoonright \varphi_e^B(e)[t] = B_t \upharpoonright \varphi_e^B(e)[t]). \tag{2.7}$$

First use S to compute $S \upharpoonright e$. For $i < e$, if $S(i) = 1$ then we will want to eventually stop enumerating markers $\lambda(i,s)$. If $S(i) = 0$, then we will want to enumerate all unrestrained markers. Suppose we see a computation $\Phi_e^B(e)[t]\downarrow$. We find the markers $\lambda(i,s)$ below $\varphi_e^B(e)[t]$ for $i < e$. As we know the fate of every marker below the use, we can computably determine whether this computation is B-correct, that is, whether B will change below the use after stage t. Therefore equation 2.7 is Σ_1^0, and can be decided by $\varnothing'$. As $\varnothing' \leqslant_T S$, we have $B' \leqslant_T S$.

2.4.3 Degrees of left-c.e. partial integer-valued randoms

Here we show that every left-c.e. real that is integer-valued random is Turing (and in fact, weak truth table) equivalent to a partial integer-valued random left-c.e. real. Hence along with the argument of the previous section, it proves Theorem 2.1.6. In order to make the argument more concise, we will often refer to the argument of Section 2.4.1, which uses a similar machinery. Given a left-c.e. integer-valued random set A we will construct a partial integer-valued random set B such that $A \equiv_{\mathrm{wtt}} B$. Suppose we are given A with a computable approximation $\langle A_s \rangle$. Let $\langle \varphi_e \rangle$ be an effective enumeration of all partial computable integer-valued martingales. We build a set B and weak truth-table reductions Γ and Δ such that $\Gamma^A = B$ and $\Delta^B = A$ to meet the requirements

R_e**:** φ_e does not succeed on B.

We also build for each $e \in \omega$ a partial integer-valued martingale m_e. In the case that φ_e is defined along B, m_e will be total.

Strategy for the single requirement R_0.

Let $\gamma(n)[s]$ be the use of computing $\Gamma^A(n)[s]$ and $\delta(n)[s]$ the use of computing $\Delta^B(n)[s]$. We begin by setting $\gamma(n)[0] = n$ and $\delta(n)[0] = n$ for all n. We observe the values of φ_0 along B. First we wait to see $\varphi_0(\lambda)$. If we later see φ_0 increase its capital along B, then we will wish to change B to force φ_0 to lose capital. We will need permission from A to do so. We put pressure on A to change by defining the martingale m_0. If φ_0 increases its capital on B_s by betting on $B_s \upharpoonright n$, then we define m_0 to start with capital $\varphi_0(\lambda)$, place the same bets as $\varphi_0[s]$ along $A_s \upharpoonright n$, and bet neutrally on all other strings up to length n. We repeat this every time we see φ_0 increase its capital along B until we see a change in A. As A is integer-valued random, $m_0(A) < \infty$ and so A must eventually move. This gives us a permission to change B.

Suppose that at stage s we have defined m_0 up to length d, and A changes below d. Let m be least such that $A_{s-1}(m) \neq A_s(m)$. We have defined $\delta(m)[s-1] = m$ and so we must change B on its mth bit. We let $B_s = B_{s-1} \restriction m\hat{}(1-B_{s-1}(m))\hat{}0^\omega$. The partial martingale φ_0 might not be defined on any string extending $B_s \restriction m$, whereas m_0 has been defined to be neutral on all initial segments of B_s of length between m and d. Later φ_0 might increase its capital along these strings, and we would not be able to define m_0 to directly copy its bets. We can however raise the use $\gamma(m)[s]$ to be large. At stage s we let $\gamma(m)[s] = 2d$. If φ_0 then bets along initial segments of B of length between m and d, we copy the wagers that φ_0 makes on these strings by placing the same wagers along the initial segments of A of length between d and $2d$. We are then still putting pressure on A to change. If A changes below $2d$ we can then change B below d and force φ_0 to lose money.

Multiple requirements

The interaction between multiple requirements will cause difficulty in coding. We use levels $\langle l_i \rangle$ and $\langle d_i \rangle$ in order to facilitate the coding. We set $\gamma(l_i) = d_i$ and $\delta(d_i) = l_{i+1}$ and let $l_1 = 5$ (the choice of l_1 is not significant). We attempt to change B above a string of length l for $l \in (l_i, l_{i+1}]$ only for the sake of the requirements $R_0, \ldots, R_i$.

We will attempt to change B below l_1 only for the sake of decreasing φ_0's capital along $B \restriction l_1$. We would like $\gamma(l_1)$ to be large enough so that we can copy all the wagers that φ_0 may place along strings of length l_1. There are 2^{l_1} many such strings, and so if we set $d_1 = \gamma(l_1) = 2^{l_1}.l_1$, this will certainly be large enough. As A is left-c.e., A can change below d_1 at most 2^{d_1} many times. Therefore there are at most 2^{d_1} many forbidden strings. To calculate l_2, we begin by setting $l_{2,0} = l_1 + (\mu x)(2^x \geqslant 2^{d_1}) = l_1 + d_1$. We act at level l_2 for the sake of R_0 and R_1. The action for these requirements can again move us closer to forbidden strings. The distance we can be moved, is given in terms of the function d which is introduced in the argument of Section 2.4.1. Therefore we define $l_{2,1}, l_{2,2}$ and l_2 as before.

Suppose A changes below d_1 at stage s. Then we are free to change B below l_1. We choose a string τ of length l_1 which minimises m_0; if there is no reason to move, we do not move. In either case, we then choose some string ρ of length l_2 extending the current version of $B \restriction l_1$ which has not been visited before, and let $B_{s+1} = \rho 1^\omega$.

We change B below l_2 for the sake of requirements R_0 and R_1. We define the total martingale m_0 to copy the wagers that φ_0 places on $B \restriction l_2$, and we define the total martingale m_1 to copy the wagers that φ_1 places on $B \restriction l_2$. We require d_2 to be large enough so that m_0 can copy all the wagers that φ_0 may place along strings of length l_2 beneath d_2, and m_1 can copy all the wagers that φ_1 may place along strings of length l_2 beneath d_2. Therefore, by the same reasoning as the calculation of d_1, we would like d_2 to be at least $2^{l_2}.l_2$. We then define l_3 similarly, with $l_{3,0} = l_2 + d_2$, and $l_{3,1}, l_{3,2}, l_{3,3}$ and l_3 as in the argument of Section 2.4.1.

Now suppose that A changes between d_1 and d_2 at stage s'. That is, $A_{s'} \upharpoonright d_1 = A_{s'-1} \upharpoonright d_1$, but there is $m \in [d_1, d_2)$ with $A_{s'}(m) \neq A_{s'-1}(m)$. We cannot change B below l_1, but we can change B above l_1. We therefore choose a string τ of length l_2 which minimises the martingales φ_0 and φ_1. As R_0 has stronger priority than R_1, we first look to minimise φ_0. If we can, we change B to minimise φ_0, and if we cannot, we then look to minimise φ_1. If we can, we change B to minimise φ_1, and if we cannot, we do not change B. In any case, we then choose some string ρ of length l_3 extending the current version of $B \upharpoonright l_2$ which has not been visited before, and let $B_{s'+1} = \rho 1^\omega$.

Construction

For every $e \in \omega$ and at every stage s we have a restraint $r_{e,s}$. During the construction we will say that *"m_e has copied φ_e's wager on σ"* for some e and string σ. Let $d_{e,s}$ denote the length of longest string for which we have defined m_e by the beginning of stage s. Set $l_0 = 0$, $d_0 = 0$ and $l_1 = 5$. Given l_i, we set $d_i = 2^{l_i}.l_i$, and then given d_i, we set $l_{i+1,0} = l_i + d_i$ and for $0 \leqslant j \leqslant i$,

$$l_{i+1,j+1} = l_{i+1,j} + \max_{k \leqslant i} n(2^{l_{i+1,j}+1} m_k(\lambda)),$$

and let $l_{i+1} = l_{i+1,i+1} + 1$. Set $\gamma(l_i) = d_i$ and $\delta(d_i) = l_{i+1}$ for all i. At stage 0, we set $B_1 = 1^\omega$, $m_e(\lambda) = \varphi_e(\lambda)$ for all e, and $r_{e,1} = l_e$ for all e. At stage $s > 0$ do the following:

Case 1: there is $l \geqslant r_{e,s}$ such that $\varphi_e(B_s \upharpoonright (l+1)) > \varphi_e(B_s \upharpoonright l)$, m_e has copied φ_e's wagers on $B_s \upharpoonright 1, \ldots, B_s \upharpoonright l$, and m_e has not already copied φ_e's wager on $B_s \upharpoonright (l+1)$. Let e be the least applicable, and l the least applicable for this e. Define m_e to wager $\varphi_e(B_s \upharpoonright (l+1)) - \varphi_e(B_s \upharpoonright l)$ on $A_s \upharpoonright (d_{e,s}+1)$ and wager 0 on all other strings of length $d_{e,s}+1$. We say that m_e has copied φ_e's wager on $B_s \upharpoonright (l+1)$. Let i be such that $l \in (l_i, l_{i+1}]$. For all $e' > e$ with $r_{e',s} \leqslant l_{i+1}$, let $r_{e',s+1} = l_{i+2}$. Proceed to the next stage.

Case 2: $A_s(m) \neq A_{s+1}(m)$. Let m be the least applicable, and let i be such that $m \in (d_i, d_{i+1}]$. Choose the least e with $r_{e,s} < l_{i+1}$ such that there is τ of length l_{i+1} extending $B_s \upharpoonright l_i$ with $\max_{j \leqslant l_{i+1}} \varphi_{e,s+1}(\tau \upharpoonright j) < \max_{j \leqslant l_{i+1}} \varphi_{e,s+1}(B_s \upharpoonright j)$. For this e, choose an applicable string τ with $\min_\mu d(\tau, \mu)$ as large as possible, where the minimum is taken over all forbidden strings of length l_{i+1} extending $B_s \upharpoonright l_i$. If there is no such e, then let $\tau = B_s \upharpoonright l_{i+1}$. Choose a string ρ of length l_{i+2} extending τ such that $\rho \nprec B_t$ for all $t \leqslant s$. Let $B_{s+1} = \rho 1^\omega$. Proceed to the next stage.

If neither case applies, proceed to the next stage.

Verification of the construction

First, we observe that $A \equiv_{wtt} B$. Indeed, the uses $\gamma(l_i)$ are clearly computable, and so we have $\Gamma^A = B$ via the weak truth-table functional Γ. The consistency of Δ is a consequence of B never extending a forbidden string. Again the uses $\delta(d_i)$ are computable and so $\Delta^B = A$ via the weak truth-table functional Δ.

It remains to show that for all $e \in \omega$, R_e is satisfied. Suppose by induction that all requirements of stronger priority than R_e do not act after stage s^*. We show that if φ_e succeeds on B, then m_e succeeds on A, which is a contradiction to A being integer-valued random.

Let i_0 be least such that $l_{i_0} \geqslant r_{e,s^*}$. By the restraints imposed, we cannot change B below l_{i_0} for the sake of R_e. Now suppose that at stage $s \geqslant s^*$ we see $\varphi_{e,s}(B_s \upharpoonright l) > \varphi_{e,s}(B_s \upharpoonright l_{i_0})$. Let l be the least applicable, and suppose i is such that $l \in (l_i, l_{i+1}]$. From stage s we have m_e copy φ_e's wagers, and at some stage $s' \geqslant s$ we define m_e to wager $\varphi_e(B_{s'} \upharpoonright l) - \varphi_e(B_{s'} \upharpoonright (l-1))$ on $A_{s'} \upharpoonright d_{e,s'} + 1$. If m_e copies all of the wagers that φ_e makes on strings of length less than or equal to $l-1$, then m_e is defined on strings of length at most $2^{l-1} \cdot (l-1)$. Therefore $d_{e,s'} \leqslant 2^{l-1} \cdot (l-1) < d_{i+1}$.

Suppose that at stage $t > s'$ we see A change below d_{i+1}. Then we will choose a string τ of length l_{i+1} extending $B_t \upharpoonright l_i$ with $\max_{j \leqslant l_{i+1}} \varphi_{e,t}(\tau \upharpoonright j) < \varphi_{e,t}(B_s \upharpoonright l)$. Taking the contrapositive, we see that if φ_e makes capital on B past l_{i_0}, then m_e makes capital on A. Therefore if φ_e succeeds on B then m_e succeeds on A.

2.5 C.e. degrees not containing IVRs

It is hardly surprising that there are c.e. degrees which do not contain integer-valued randoms. After all, computable enumerability hinders randomness, and indeed with respect to a sufficient level of randomness, c.e. sets are not random. However integer-valued randomness is sufficiently weak so that it has interesting interactions with computable enumerability. In this section we look at the question of which c.e. degrees do not contain integer-valued randoms. The first example of such degrees was given in Section 2.3.2 where we showed Theorem 2.1.7. Perhaps more surprising is the fact that there are c.e. degrees which are not array computable (so, by Corollary 2.1.5 they compute an integer-valued random) yet they do not contain integer-valued randoms. We prove this in Section 2.5.1, and extend it to a much stronger result (namely Theorem 2.1.8) in Section 2.5.2.

2.5.1 C.e. array noncomputable degrees not containing IVRs

We wish to construct an array noncomputable c.e. degree not containing an integer-valued random set. We use the original definition of array noncomputable from [19] in terms of very strong arrays. Let $(\Gamma_e, \Delta_e)_{e\in\omega}$ be an effective listing of all pairs of Turing functionals, and let $\langle D_n \rangle$ be the very strong array with $D_0 = \{0\}, D_1 = \{1,2\}, D_2 = \{3,4,5\}, D_3 = \{6,7,8,9\}, \ldots$. We build a c.e. set B to satisfy the requirements

R_e: $(\exists n)(W_e \cap D_n = B \cap D_n)$

N_e: $\Delta_e^B = A_e \wedge \Gamma_e^{A_e} = B \implies A_e$ is not integer-valued random.

We build for each $e \in \omega$ an integer-valued martingale m_e, and replace the requirement N_e with the following requirements $N_{e,k}$ for all $k \geqslant 2$:

$N_{e,k}$: $\Delta_e^B = A_e \wedge \Gamma_e^{A_e} = B \implies m_e$ wins at least k dollars on A_e.

We effectively order the requirements, making sure that if $k < k'$, then $N_{e,k}$ has stronger priority than $N_{e,k'}$. We say that R_e requires attention at stage s if

1. R_e has no follower at stage s, or

2. R_e has follower i at stage s and $W_{e,s} \cap D_i \neq B_s \cap D_i$.

For any $e \in \omega$, let $d_{e,s}$ be the length of the longest string σ for which $m_e(\sigma)$ is defined by stage s. We have for every $e \in \omega$ a restraint r_e. Let $r_{e,k,s} = \max D_i$ where i is the follower at stage s of any R-requirement of stronger priority than $N_{e,k}$. Let $l(e,s)$ be the length of agreement between $\Gamma_e^{\Delta_e^B}$ and B at stage s,

$$l(e,s) = \max\{x \mid (\forall y < x)(\Gamma_e^{\Delta_e^B}(y)[s] = B(y)[s]\}.$$

We say that $N_{e,k}$ requires attention at stage s if

1. $m_e(A_{e,s} \upharpoonright d_{e,s}) = k - 1$,

2. $l(e,s) > r_{e,k,s}$,

3. $l(e,s) > \delta_e(\gamma_e(r_{e,k,s}))[s]$

4. $\gamma_e(l(e,s))[s] > d_{e,s}$

Figure 2.3 illustrates the reductions involved in requirement N_e.

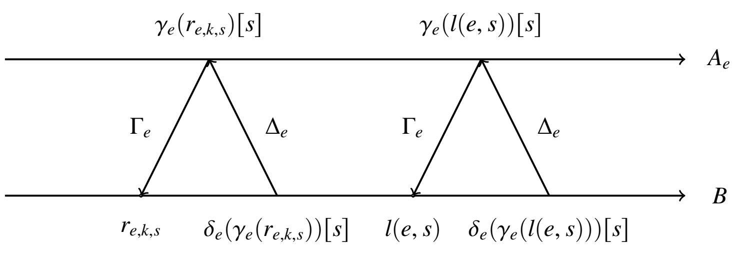

Figure 2.3: A visualization of the reductions in requirement N_e

Construction At stage 0, let $B_0 = \varnothing$. Let $m_e(\lambda) = 1$ for all $e \in \omega$. At stage s, $s \geqslant 1$, find the requirement of strongest priority which requires attention at stage s.

Case 1: this is R_e. If R_e has follower i, enumerate $W_{e,s} \cap D_i$ into B. If R_e does not have a follower, appoint a fresh large follower for R_e.

Case 2: this is $N_{e,k}$. Let $\tau = A_e[s] \upharpoonright \gamma_e(l(e,s))[s]$. Define m_e to wager \$1 on τ, and bet neutrally on all other strings with length in $(d_{e,s}, |\tau|]$. Remove the followers of R-requirements of weaker priority than $N_{e,k}$.

Verification It remains to show that each requirement requires attention only finitely often, and is met. Assume by induction that all requirements of stronger priority than R_e do not require attention after stage s. If R_e does not have a follower at stage s then it will be appointed one. This follower cannot be cancelled as requirements of stronger priority can no longer act. Suppose that R_e has follower i at stage s. If we ever see that $B_t \cap D_i \neq W_{e,t} \cap D_i$ then we will enumerate $W_{e,t} \cap D_i$ into B. As $|D_i| = i + 1$, R_e can require attention at most

$i+1$ many times after stage s. Then we will have $B \cap D_i = W_e \cap D_i$ and R_e is satisfied.

We claim that for all e, if $\Delta_e^B = A_e$ and $\Gamma_e^{A_e} = B$, then m_e is nondecreasing along A_e. Suppose we have $m_e(\sigma) = k-1$ for some string σ. Suppose $N_{e,k}$ requires attention at stage t and we define $m_e(\tau) = k$. We remove the followers for R-requirements of weaker priority, and so only requirements of stronger priority than $N_{e,k}$ can enumerate elements into B that are below $\delta_e(\gamma_e(l(e,t))[t]$. Suppose R_j has stronger priority than $N_{e,k}$. Then R_j can either enumerate elements below $r_{e,k,t}$, or if it is later injured, enumerate elements larger than $\delta_e(\gamma_e(l(e,t))[t]$ into B. Suppose that R_j enumerates an element below $r_{e,k,t}$ into B. If Γ_e and Δ_e later recover at stage t', $\tau' := A_e[t'] \upharpoonright \gamma_e(r_{e,k,t})[t]$ must be incomparable with $\tau = A_e[t] \upharpoonright \gamma_e(r_{e,k,t})[t]$; otherwise Δ_e will not have recorded the B-change and we could not have $\Delta_e^B = A_e$ and $\Gamma_e^{A_e} = B$. In particular, τ' must not extend either τ or its sibling (recall the definition of the *sibling* of a string, just after Definition 2.1.1). As m_e is defined to bet neutrally on strings of length $|\tau|$ that are not either τ or its sibling, we will have $m_e(A_e[t'] \upharpoonright |\tau|) \geqslant k-1$. By induction, this holds for all k. This establishes the claim.

Now assume by induction that all requirements of stronger priority than $N_{e,k}$ do not require attention after some stage s. As $N_{e,j}$ for any $j < k$ does not require attention, we must have $m_e(A_{e,s} \upharpoonright d_{e,s}) = k-1$. If $N_{e,k}$ does not require attention at any stage $t > s$ then the hypothesis of the requirement does not hold. Therefore $N_{e,k}$ is satisfied vacuously. If $N_{e,k}$ does require attention at stage $t > s$ then we define $m_e(\tau) = k$ for $\tau = A_e[t] \upharpoonright \gamma_e(l(e,t))[t]$. R-requirements of stronger priority have finished acting, and so no numbers less than $r_{e,k,t}$ enter B after stage t. We remove the followers for R-requirements of weaker priority. When they are appointed new followers they will choose fresh numbers, and so all enumerations into B after stage t will be larger than $\delta_e(\gamma_e(l(e,t))[t]$. As B cannot change below $\delta_e(\gamma_e(l(e,t)))[t]$, $A_e = \Delta_e^B$ cannot change below $\gamma_e(l(e,t))[t]$. Therefore $\tau \prec A_e$ and $m_e(A_e \upharpoonright \gamma_e(l(e,t))[t]) = k$. By the previous claim, $m_e(A_e) \geqslant k$, and $N_{e,k}$ is satisfied.

2.5.2 A high$_2$ c.e. degree not containing integer-valued randoms

In this section we prove Theorem 2.1.8. Nies, Stephan and Terwijn showed that every high degree contains a computably random set, and so a fortiori, an integer-valued random set. It is instructive though to see why we cannot build a high degree that does not contain an integer-valued random set. This will give us some insight as to why the construction works when we only require our set to be high$_2$. So that the degree of A does not contain an integer-valued random set, we meet the requirements

$$\mathcal{N}_e\text{: } \Gamma_e^{\Delta_e^A} = A \text{ total} \implies \Delta_e^A \text{ is not integer-valued random}$$

where (Γ_e, Δ_e) is an enumeration of pairs of Turing functionals. We break the requirement $\mathcal{N}_e$ into the following subrequirements $\mathcal{N}_{e,k}$.

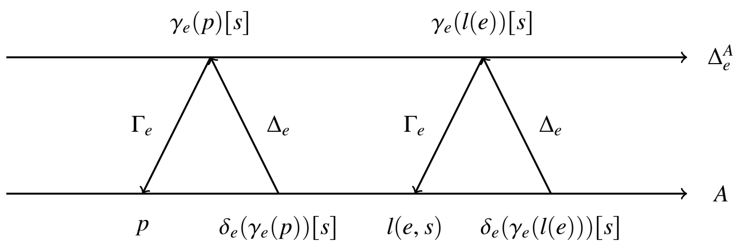

Figure 2.4: Diagram with the uses and where we bet, for the proof of Theorem 2.1.8.

$\mathcal{N}_{e,k}$: $\Gamma_e^{\Delta_e^A} = A$ total $\implies$ m_e wins at least k dollars on Δ_e^A

where m_e is an integer-valued martingale we build for the sake of $\mathcal{N}_e$. Suppose that the martingale m_e has won \$$k-1$ on $\Delta_e^A \upharpoonright n$. The basic strategy to win another dollar is to first pick a large location marker p. Let $l(e)[s]$ be the length of agreement between A and $\Gamma_e^{\Delta_e^A}$ at stage s. If we later see that $l(e)[s] > p$ and $l(e)[s] > \delta_e(\gamma_e(p))[s]$, then we define m_e to wager \$1 on $\Delta_e^A[s] \upharpoonright \gamma_e(l(e))[s]$ and bet neutrally elsewhere, and freeze A below $\delta_e(\gamma_e(l(e))[s]$. If we are successful in freezing A, then m_e wins \$1 on Δ_e^A. If A changes below p at stage $s' > s$, then if we are to have $\Gamma_e^{\Delta_e^A} = A$, $\Delta_e^A[s'] \upharpoonright \gamma_e(p)[s]$ is incomparable with $\Delta_e^A[s] \upharpoonright \gamma_e(p)[s]$ and m_e does not lose any capital along Δ_e^A. We can then try the basic strategy again.

To make A high we would define a functional Λ such that $\lim_k \Lambda^A(x,k) = \mathrm{Tot}(x)$, where Tot is the canonical Π_2^0-complete set. The basic strategy for the highness requirement is to define $\Lambda^A(x,s) = 0$ for larger and larger s with some big use $\lambda(x,s)$. When we see $\varphi_x(s') \downarrow$ for all $s' \leqslant s$, then for each $s' \leqslant s$ we enumerate the current use $\lambda(x,s')$ into A (if currently $\Lambda^A(x,s') = 0$) and redefine $\Lambda^A(x,s') = 1$ with use -1, i.e. the axiom defining $\Lambda^A(x,s') = 0$ does not depend on A.

This strategy will succeed as long as we are prevented from redefining $\Lambda^A(x, s)$ from 0 to 1 at most finitely often.

Let us see how these strategies might interact. Suppose at stage s we saw the $l(e)$ computations converge and defined more of m_e. The highness requirement, if unrestrained, can destroy the $l(e)$ computations and cause m_e to lose capital if it enumerates a marker between p and $\delta_e(\gamma_e(l(e))[s]$. Therefore if m_e is to ever win money along Δ_e^A we must impose restraint on A. The problem is that the strategies for $\mathcal{N}_e$ may gang up and impose restraint on *all* markers $\lambda(x, s)$; every time a marker is defined we may define m_e and impose restraint, and never allow the marker to be enumerated. If $\mathrm{Tot}(x) = 1$ we will never be able to correct $\Lambda^A(x, s)$ to be 1 and the limit will be incorrect.

Another approach we might take would be to capriciously enumerate the markers which occur below $\delta_e(\gamma_e(l(e))[s]$. If we do this and always have some marker below the use, we will be able to conclude that $\Gamma_e^{\Delta_e^A} \neq A$ (this argument is given in full in the verification below). However, the problem now is that we might not have wanted to enumerate the markers. If $\mathrm{Tot}(x) = 0$ and we capriciously enumerate all markers $\lambda(x, s)$ and redefine $\Lambda^A(x, s)$ to be 1, the limit will be incorrect.

To make A high$_2$, we need to instead define a functional Λ such that

$$\lim_m \lim_t \Lambda^A(x, m, t) = \mathrm{Cof}(x)$$

where $\mathrm{Cof} = \{x \mid W_x \text{ is cofinite}\}$ is the canonical Σ_3^0-complete set.

The double limit means that we may be wrong on a finite number of the m while still satisfying the requirement. This will allow us to employ the capricious enumeration strategy successfully. We have the requirements

$\mathcal{H}_x$**:** $\lim_m \lim_t \Lambda^A(x, m, t) = \mathrm{Cof}(x)$

as well as the $\mathcal{N}_e$ from above. The construction will use a priority tree. For each global requirement $\mathcal{N}_e$ we have several nodes devoted to meeting $\mathcal{N}_e$, each equipped with a guess as to the outcomes of stronger priority requirements. Such a node will be called a *mother node* devoted to $\mathcal{N}_e$. Each such node τ builds its own martingale m_τ. We argue in the verification that for every $e \in \omega$ there is some node τ such that m_τ succeeds on Δ_e^A. For each subrequirement $\mathcal{N}_{e,k}$ we have several nodes devoted to meeting $\mathcal{N}_{e,k}$. Such a node will be called a *worker node* devoted to $\mathcal{N}_{e,k}$, and will occur below a mother node τ devoted to $\mathcal{N}_e$. For the longest such τ, we say that σ's mother node is τ. When we reach a node σ devoted to $\mathcal{N}_{e,k}$, we choose a fresh location marker p for σ, and place a *link* from σ back up to τ. The link can be seen as testing the hypothesis of $\mathcal{N}_e$. The length of agreement between A and $\Gamma_e^{\Delta_e^A}$ will be measured at τ. When we next arrive at τ and see the length of agreement computations converge, this further confirms the hypothesis of $\mathcal{N}_e$. We travel the link to σ, define more of the martingale m_τ, and then remove the link.

The requirement $\mathcal{H}_x$ will have nodes $\beta_{x,m}$ for $m \in \omega$. The node $\beta_{x,m}$ tests whether $[m, \infty) \subseteq W_x$. Note that this is a Π_2 test. Such a node will have outcomes ∞, which corresponds to the Π_2 test infinitely often looking correct, and f, corresponding to the finite outcome. The $\beta_{x,m}$ nodes will be responsible for defining $\Lambda^A(x,m,t)$ for each t. As we do not know the true path in advance, each path through the priority tree will have a $\beta_{x,m}$ node, which collectively will define $\Lambda^A(x,m,t)$ for all t.

The basic strategy for $\beta_{x,m}$ is to define $\Lambda^A(x,m,t) = 0$ for larger and larger t with some big use $\lambda(x,m,t)$. If we see $[m,s] \subseteq W_x$ then for each $s' \leqslant s$ we enumerate the current use $\lambda(x,m,s')$ into A (if currently $\Lambda^A(x,m,s') = 0$) and redefine $\Lambda^A(x,m,s') = 1$ with use -1, i.e. the axiom defining $\Lambda^A(x,m,s') = 0$ does not depend on A. It is important to note that the markers $\lambda(x,m,s)$ are *shared* by all the $\beta_{x,m}$ nodes. Whether we succeed in enumerating the markers and updating Λ as needed will depend on how the construction proceeds.

We will now describe how these requirements interact and what modifications we need to make to the priority tree as a consequence. First we consider the situation where we have $\mathcal{H}_x$ of lower priority than $\mathcal{N}_e$ (which is associated with the mother node τ). The problem is the following. Suppose we have a situation with nodes $\tau \prec \beta_{x,m} \prec \sigma$ where σ is devoted to $\mathcal{N}_{e,k}$ and τ is σ's mother node. That is, while σ has higher *global* priority than $\beta_{x,m}$, its *local* priority is lower. Suppose at some stage σ picks a location marker $p(\sigma)$ and creates a link back to τ at stage s_0. At a later stage s_1 we get to τ, see the necessary computations converge, and would like to travel the link and define the martingale. This causes no problem if

$\sigma \not\geqslant \beta_{x,m}\hat{\ } f$, but there are problems if $\sigma \not\geqslant \beta_{x,m}\hat{\ }\infty$. We will require the computations $l(\tau)[s]$ to be τ-correct; that is, all guesses τ makes about the enumeration of markers below $\delta_e(\gamma_e(l(e)))[s]$ have already occurred.

The trouble is that at stage s_1 there now might be some marker $\lambda = \lambda(x, m, q)$ which is greater than $p(\sigma)$, but below the use of $l(\tau)[s_1]$. We may not yet want to enumerate λ into A because the Σ_2 outcome may now be looking correct at $\beta_{x,m}$ (that is, we might think $\mathrm{Cof}(x) = 0$). If we did define the martingale, since $\beta_{x,m}$ has higher priority than σ, any restraint imposed at s_1 may not be successful since $\beta_{x,m}$ might later put λ into A. This could potentially cause our martingale m_τ to lose all its capital, and it could never bet again.

The solution to this problem is as follows. When we hit τ, if there is some link to a node σ and there is some λ as above, we immediately enumerate any λ below the use of $l(\tau)[s]$ into A, but we do *not* define the martingale. This means that $\beta_{x,m}$ cannot later use λ to make m_τ lose capital. If there is no such λ then we do define the martingale, since we can be sure that σ is satisfied provided that it is on the true path. This is the situation we would like, but failing that, we would like to get a global win on $\mathcal{N}_e$. In the case that such a λ exists, we travel the link from τ to σ, enumerate all applicable markers, but we do *not* delete the link. Because of this, we need to add a new outcome to σ. Therefore σ will have outcomes g and d. The outcome g will be played when we perform the capricious enumeration of λ as above. The outcome d will be played when we define the martingale. Suppose we have some worker node σ with location marker $p(\sigma)$ which always has some λ below the use of $l(\tau)[s]$. We will then define $\Lambda^A(x, m, t)$ to have limit 1 for any pair (x, m) such that $\tau \prec \beta_{x,m} \prec \sigma$; note that there are only finitely many pairs (x, m). We will also have a link from τ to σ for almost all stages. This corresponds, however, to a global win on $\mathcal{N}_e$, since $p(\sigma)$ is a witness to the fact that $\delta_e(\gamma_e(l(e)))$ does not exist and so $\Gamma_e^{\Delta_e^A} \neq A$. The permanent link may cause us to skip over other mother nodes, which would mean we cannot meet their requirements. We therefore *restart* all $\mathcal{N}$-requirements of weaker priority than $\mathcal{N}_e$ under the g outcome of σ. We do this by assigning the requirements $\mathcal{N}_{e'}$ for $e' > e$, as well as their subrequirements $\mathcal{N}_{e',k}$ for $k \in \omega$, to nodes below $\sigma\hat{\ }g$ in some fair way. We do not restart any β nodes, since $\Lambda^A(x, m, t)$ will be defined to be 1 for the finitely many x and m with $\tau \prec \beta_{x,m} \prec \sigma$. This will mean that for a finite number of m, $\lim_t \Lambda^A(x, m, t)$ may be incorrectly outputting 1 instead of 0. This is fine though, since we will only lose on a finite number of the m and still can satisfy $\mathcal{H}_x$. The sacrifice of losing on an m will only be made when we can ensure a global win on a τ node of stronger priority.

We now come to the situation where we have an $\mathcal{H}_x$ of *higher* global priority than the $\mathcal{N}_e$ associated with τ. We now cannot allow τ to capriciously enumerate all the markers belonging to $\beta_{x,m}$ if f is $\beta_{x,m}$'s true outcome. We now describe our solution to this problem.

First suppose that $\beta_{x,m}$ and $\beta_{x,n}$ are worker nodes devoted to $\mathcal{H}_x$

with $m < n$. If $\beta_{x,n}$ occurs below $\beta_{x,m}\widehat{\ }\infty$, then $\beta_{x,n}$ is guessing that $[m,\infty) \subseteq W_x$. Therefore $\beta_{x,n}$ must also be guessing that $[n,\infty) \subseteq W_x$ as $n > m$, and so $\beta_{x,n}$ will have only the ∞ outcome.

Suppose that τ is below the f outcome of any $\beta_{x,m}$ node with $m < n$. We will restart τ below $\beta_{x,n}\widehat{\ }\infty$. Consider the situation $\beta_{x,n}\widehat{\ }\infty \preccurlyeq \tau \prec \beta_{x,n'} \prec \sigma$ with $n' > n$. If there is a link from τ to σ then capriciously enumerating the markers $\lambda(x,n',t)$ into A will not injure $\beta_{x,n'}$ since this is what $\beta_{x,n'}$ would like to do anyway. Therefore $\mathcal{H}_x$ cannot be injured in this situation. We show in Lemma 2.5.1 that such a τ can be restarted only finitely many times.

There is one last problem. Suppose we have $\tau \prec \beta_{x,m} \prec \sigma$ with a link (τ,σ). As σ's mother is above $\beta_{x,m}$, we must have $\beta_{x,m}\widehat{\ }f \preccurlyeq \sigma$. If the link is permanent then $\beta_{x,m}$ will not be able to enumerate its markers. As $\mathcal{H}_x$ has higher global priority than $\mathcal{N}_e$, this is not a situation we want. We employ the following technique from Downey-Stob [22]. When we hit τ, we realize that if there is a link from τ down then this may be a potentially permanent link. We first perform a *scouting report* to see where we would go if there were no link around. If we were to go to a node γ to the left of σ then we will erase the link and actually go to γ instead. This ensures that if $\beta_{x,m}\widehat{\ }\infty$ is $\beta_{x,m}$'s true outcome, then it will be able to enumerate its markers.

The Priority Tree

Our priority tree, PT, will have three types of nodes. The first type are *mother* nodes τ, which have outcomes ∞ and f, and will be assigned to some global requirement $\mathcal{N}_e$. We write $e(\tau) = e$. The next type are *worker* nodes σ which are devoted to a subrequirement of some $\mathcal{N}_e$, and hence will be assigned some e, k. We write $e(\sigma) = e, k(\sigma) = k$. We form the tree so that such σ occur below some τ with $e(\tau) = e$. For the longest such τ with $e(\tau) = e$, we will write $\tau(\sigma) = \tau$. This is to indicate that τ is σ's mother. σ has outcomes g and d with $g < d$. Finally we have nodes β which are devoted to some $\mathcal{H}_x$, and hence will be assigned some x, m. We write $x(\beta) = x, m(\beta) = m$, or simply $\beta_{x,m}$. β will have outcomes ∞ and f, with $\infty < f$, unless β occurs below $\beta' \hat{} \infty$ for some β' with $x(\beta') = x(\beta)$, in which case β has only the single outcome ∞.

We now assign requirements and subrequirements to nodes on the tree. In a basic infinite injury argument we would have all nodes of the same level working for the same requirement. However in our case, as we must restart τ nodes, it is more complicated. We use *lists* of, for example, Chapter XIV of [43] for this. We will have three lists, L_0, L_1, and L_2, which keep track of indices for τ, σ and β nodes, respectively.

$n = 0$. Let λ be devoted to $\mathcal{N}_0$, and let $L_0(\lambda) = L_1(\lambda) = L_2(\lambda) = \omega$.

For $n > 0$, let $\gamma \in PT$ be of the form $\delta \hat{} a$. Adopt the first case below to pertain, letting $L_i(\gamma) = L_i(\delta)$ unless otherwise mentioned.

Case 1. δ is devoted to $\mathcal{N}_e$.

Case 1a. $a = f$. $L_0(\gamma) = (L_0(\delta) - \{e\}) \cup \{e' \mid e' > e\}$

$$L_1(\gamma) = (L_1(\delta) - \{\langle e,k\rangle \mid k \in \omega\}) \cup \{\langle e',k\rangle \mid e' > e, k \in \omega\}.$$

Case 1b. $a = \infty$. $L_0(\gamma) = L_0(\delta) - \{e\}$.

Case 2. δ is devoted to $\mathcal{N}_{e,k}$.

Case 2a. $a = g$. Define the lists as in Case 1a.

Case 2b. $a = d$. Let $L_1(\gamma) = L_1(\delta) - \{\langle e,k\rangle\}$.

Case 3. δ is devoted to $\mathcal{H}_x$ with $m(\delta) = m$.

Case 3a. $a = f$. $L_2(\gamma) = L_2(\delta) - \{\langle x,m\rangle\}$.

Case 3b. $a = \infty$. Let $L_0(\gamma)$ be the union of $L_0(\delta)$ with

$$\{e \mid (\exists\tau)(\forall\beta)(e(\tau) = e \wedge x(\beta) = x \wedge x < e \wedge \beta \prec \tau \prec \delta \implies \beta\hat{\ } f \preccurlyeq \tau)\}$$

and let $L_1(\gamma)$ be the union of $L_0(\delta)$ with

$$\{\langle e,k\rangle \mid (\exists\tau)(\forall\beta)(e(\tau) = e \wedge x(\beta) = x \wedge x < e \wedge \beta \prec \tau \prec \delta \implies \beta\hat{\ } f \preccurlyeq \tau), k \in \omega\}.$$

Also let $L_2(\gamma) = L_2(\delta) - \{\langle x,m\rangle\}$.

Having defined the lists, we now assign requirements to nodes of the priority tree as follows. Let $\gamma \in \mathrm{PT}$ and i be the least element of $L_0(\gamma) \cup L_1(\gamma) \cup L_2(\gamma)$. If $i \in L_0(\gamma)$, let γ be a mother node devoted to $\mathcal{N}_i$. If $i \in L_1(\gamma) - L_0(\gamma)$ and $i = \langle e,k\rangle$, let γ be a worker node devoted to $\mathcal{N}_{e,k}$. Otherwise $i = \langle x,m\rangle \in L_2(\gamma)$ and we let γ be worker node devoted to $\mathcal{H}_x$ with $m(\gamma) = m$.

Lemma 2.5.1 (Finite injury along any path lemma)**.** *For every path $h \in [PT]$ and every $e,k \in \omega$,*

1. $(\exists^{<\infty}\alpha \prec h)(e(\alpha) = e \wedge h(|\alpha|) = g)$,

2. $(\exists^{<\infty}\alpha \prec h)(\alpha$ *devoted to* $\mathcal{N}_e)$,

3. $(\exists^{<\infty}\alpha \prec h)(\alpha$ *devoted to* $\mathcal{N}_{e,k})$.

Proof. (1) and (3) are straightforward. For (2), a node τ devoted to $\mathcal{N}_e$ is restarted below $\beta\hat{\ }\infty$ if $x(\beta) < e$ and τ has been below only the f outcomes of nodes devoted to $\mathcal{H}_x$. Once restarted, it can no longer be restarted below any other β' with $x(\beta') = x(\beta)$. Thus τ is restarted finitely many times. □

The construction below will proceed in substages. We will append a subscript t to a parameter G, so that G_t denotes the value of G at substage t of the construction. As usual all parameters hold their value unless they are initialized. When initialization occurs they become undefined, or are set to zero as the case may be. We will append a parameter $[s]$, when necessary, to denote stage s. We may write (s,t) to denote substage t of stage s.

If we visit a node ν at stage (s,t) we will say that (s,t) is a genuine ν-stage. It might be that we do not visit ν at stage (s,t), rather we visit some ν' extending ν. In this case we say that (s,t) is a ν-stage, and hence a ν-stage may not be genuine. In fact, should we put in place some permanent link (τ,σ) with $\tau \prec \nu \prec \sigma$, then ν might only ever be visited finitely often. However, this is when $\sigma\hat{\ }g$ is the true outcome for some higher priority τ, and we would claim that a new version of ν would live below outcome g of σ. We will eventually define the genuine true path as those nodes that are on the leftmost path visited infinitely often, and for which there are infinitely many genuine stages.

We have for each node on the priority tree γ and for all $x, m \in \omega$ a restraint $c_\gamma(x, m)$. These restraints will be initially set to zero in the construction, and will only be increased when γ is a mother node devoted to some $\mathcal{N}_e$. We say that a computation $\Xi^A(x)[s]$ is τ-correct if $(\forall q \leqslant s)(\forall \beta)$ the condition

$$(\beta\hat{\ }\infty \leqslant \tau \wedge x(\beta) = x \wedge m(\beta) = m \wedge \max_{\tau' \leqslant \beta} c_{\tau'}(x, m)[s] < \lambda(x, m, q) < \xi^A(x)[s])$$

implies $\lambda(x, m, q) \in A[s]$. If τ is a mother node devoted to $\mathcal{N}_e$, then let

$$l_1(\tau)[s] = \max\{x \mid (\forall y < x)(\Gamma_e^{\Delta_e^A}(y)[s] = A[s](y)) \text{ via a } \tau\text{-correct computation}\}.$$

Let $m_1(\tau)[s] = \max\{l_1(\tau)[s'] \mid s' < s \text{ is a genuine } \tau\text{-stage}\}$. If τ is a node devoted to $\mathcal{H}_x$ with $m(\tau) = m$, then let $l_2(\tau)[s] = \max\{y \mid [m, y] \subseteq W_x[s]\}$. Let

$$m_2(\tau)[s] = \max\{l_2(\tau)[s'] \mid s' < s \text{ is a genuine } \tau\text{-stage}\}.$$

For τ a mother node devoted to some $\mathcal{N}_e$, let $d_\tau[s]$ denote the length of the longest string for which m_τ is defined by stage s.

Construction At stage 0 set $\Lambda^A(0, 0, 0) = 0$ with choose some use $\lambda(0, 0, 0)$. Set $c_\gamma(x, m)[0] = 0$ for all $x, m \in \omega$ and all nodes γ on the priority tree. Set $m_\tau(\lambda) = 1$ for all nodes τ devoted to some requirement $\mathcal{N}_e$. Stage $s + 1$ will proceed in substages $t \leqslant s$. As usual, we will generate a set of accessible nodes, $\mathrm{TP}[s+1]_t$, and will automatically initialize nodes α to right of $\mathrm{TP}[s+1]_t$. A node is initialized by removing its location marker and removing any link to or from the node.

Substage 0. Define $\mathrm{TP}[s+1]_0 = \lambda$, the empty string. Let $\Lambda^A(x,m,s+1) = 0$ with some large use $\lambda(x,m,s+1)$ for all $x,m \leqslant s+1$.

Substage $t+1 \leqslant s+1$. We will be given a string $\gamma = \mathrm{TP}[s+1]_t$. Adopt the first case to pertain below.

Case 1. γ is a mother node devoted to $\mathcal{N}_e$.

Subcase 1a. There is a link (γ,σ) for some node σ. We perform the *scouting report* by computing the string γ' that would be $\mathrm{TP}[s+1]$ were there no link. If $\gamma' <_L \sigma$, remove the link (γ,σ), let $\mathrm{TP}[s+1]_{t+1} = \gamma' \upharpoonright (|\gamma|+1)$, and go to substage $t+2$. If $\gamma' \not<_L \sigma$, see whether $l_1(\gamma)[s+1] > p(\sigma), \delta_e(\gamma_e(p(\sigma)))[s+1]$ with $\gamma_e(p(\sigma))[s+1] > d_\gamma[s+1]$.

Subcase 1a.1. No. Set $\mathrm{TP}[s+1]_{t+1} = \tau\hat{\ }f$ and go to substage $t+2$.

Subcase 1a.2. Yes and for some node β devoted to $\mathcal{H}_x$ with $m(\beta) = m$ and $\gamma \prec \beta\hat{\ }\infty \preccurlyeq \sigma$, there is a marker $\lambda(x,m,q)$ with $p(\sigma) \leqslant \lambda(x,m,q) \leqslant \delta_e(\gamma_e(l_1(\gamma)))[s+1]$. Our action is to set $\mathrm{TP}[s+1]_{t+1} = \sigma$. We refer to this action as traveling the link. Go to substage $t+2$.

Subcase 1a.3. Otherwise, set $\mathrm{TP}[s+1]_{t+1} = \sigma$ and go to substage $t+2$.

Subcase 1b. There is no link from γ. Let $\mathrm{TP}[s+1]_{t+1} = \gamma\hat{}\infty$.

Case 2. γ is a worker node devoted to $\mathcal{N}_{e,k}$.

Subcase 2a. We were in subcase 1a.2 in the previous substage. Enumerate all markers as in the previous substage into A. Let $\mathrm{TP}[s+1]_{t+1} = \gamma\hat{}g$.

Subcase 2b. We were in subcase 1a.3 in the previous substage. Define $m_{\tau(\gamma)}$ to wager \$1 on $\Delta_e^A[s+1] \upharpoonright \gamma_e(l_1(\tau(\gamma)))[s+1]$ and bet neutrally on all other strings up to and including that length. For all x, m such that there is $\beta \succ \gamma$ with $x(\beta) = x, m(\beta) = m$, let $c_{\tau(\gamma)}(x,m)[s+1] = \delta_e(\gamma_e(l_1(\tau(\gamma)))[s+1]$. Remove the link $(\tau(\gamma),\gamma)$. Let $\mathrm{TP}[s+1]_{t+1} = \gamma\hat{}d$.

Subcase 2c. We did not travel a link to arrive at γ. If $m_{\tau(\gamma)}(A_s \upharpoonright d_{\tau(\gamma)}[s+1]) \geqslant k$, let $\mathrm{TP}[s+1]_{t+1} = \gamma\hat{}d$, and go to substage $t+2$. If not, choose a fresh large follower $p(\gamma)$ for γ, place a link $(\tau(\gamma),\gamma)$, and go to stage $s+2$.

Case 3. γ is a node devoted to $\mathcal{H}_x$ with $m(\gamma) = m$. Consider the immediate successors of γ on the priority tree.

Case 3a. The immediate successors of γ are $\gamma\hat{}\infty$ and $\gamma\hat{}f$. See whether $l_2(\gamma)[s+1] > m_2(\gamma)[s+1]$.

Case 3a.1. Yes. For all $q \leqslant m_2(\gamma)[s+1]$, if $\lambda(x,m,q) > \max_{\tau \leqslant \gamma} c_\tau(x,m)[s+1]$ and $\Lambda^A(x,m,q) = 0$, enumerate $\lambda(x,m,q)$ into A and define $\Lambda^A(x,m,q) = 1$ with use -1. Set $\mathrm{TP}[s+1]_{t+1} = \gamma\hat{}\infty$.

Case 3a.2. No. Set $\mathrm{TP}[s+1]_{t+1} = \gamma\hat{}f$.

Case 3b. The immediate successor of γ is $\gamma\hat{\ }\infty$. For all $q \leqslant m_2(\gamma)[s+1]$, if

$$\lambda(x,m,q) > \max_{\tau\leqslant\gamma} c_\tau(x,m)[s+1] \text{ and } \Lambda^A(x,m,q) = 0$$

enumerate $\lambda(x,m,q)$ into A and define $\Lambda^A(x,m,q) = 1$ with use -1. Set $\mathrm{TP}[s+1]_{t+1} = \gamma\hat{\ }\infty$.

Verification We define TP, the *true path*, to be the leftmost path visited infinitely often. This clearly exists as the priority tree is finite-branching. We define GTP, the *genuine* true path, to be those $\alpha \prec \mathrm{TP}$ such that there are infinitely many *genuine* α-stages.

Lemma 2.5.2. *For every $e \in \omega$ there is a node τ devoted to $\mathcal{N}_e$ on GTP.*

Proof. Let τ be the longest node devoted to $\mathcal{N}_e$ on TP, which exists by the finite injury along any path lemma. We claim that τ is on GTP. Suppose otherwise. Then it must be the case that there are τ' and σ on GTP such that $\tau' \prec \tau \prec \sigma$ and the link (τ', σ) is there at almost all stages. This implies that $\sigma\hat{\ }g$ is on GTP. On the priority tree, if $\tau_1 \prec \tau_2$ then $e(\tau_1) < e(\tau_2)$. Now as σ links to its mother node τ' and $\tau' \prec \tau$ we must have $e(\tau') < e(\tau) = e$. But then by the construction of the priority tree, there is an $\mathcal{N}_e$ node below $\sigma\hat{\ }g$, contradicting the hypothesis that τ is the longest such. $\square$

Lemma 2.5.3. *For every $e \in \omega$, $\mathcal{N}_e$ is satisfied.*

Proof. Let τ be the longest node on GTP devoted to $\mathcal{N}_e$. First suppose that $\tau\hat{\ }f <$ GTP. Then $\Gamma_e^{\Delta_e^A} \neq A$ and $\mathcal{N}_e$ is vacuously satisfied. If there is a permanent link (τ, σ) for some node σ, then we claim that $\Gamma_e^{\Delta_e^A} \neq A$. For contradiction suppose $\Gamma_e^{\Delta_e^A} = A$, and suppose the link was placed at stage s_0. Then there is a stage $s > s_0$ and uses $\delta_e(\gamma_e(l_1(\tau))) = a_1, \gamma_e(l_1(\tau)) = a_2$, such that $\Gamma_e^{\Delta_e^{A\restriction a_1}\restriction a_2}[s] \preccurlyeq A \restriction a_1$. However if this were the case, at the next genuine τ-stage greater than s we will see that these computations have converged, play the d outcome, and remove the link. This contradicts the fact that the link is permanent. This establishes the claim.

Now suppose there is no such permanent link. We will show that m_τ succeeds on Δ_e^A. We must ensure that m_τ's capital does not decrease along Δ_e^A. Fix $k \in \omega$, and let $\sigma' > \tau$ be the first node devoted to $\mathcal{N}_{e,k}$ that is visited. Suppose we visit σ' first at stage s_0. At stage s_0 we assign σ' a fresh large location marker $p(\sigma')$ and link back to τ. Let s_1 be the stage at which we first define m_τ to win \$$k$ on $\Delta_e^A[s_0]$. As we acted in case (2b) of the construction, we did not play the g outcome at stage s_1 and so there were no markers belonging to any nodes β such that $\tau < \beta\hat{\ }\infty \preccurlyeq \sigma'$ below the use of our computations. The computations are τ-correct at stage s_1 and restraint is imposed on requirements of weaker priority than σ'. Therefore the only markers which can be enumerated below the use are those belonging to nodes β such that $\tau < \beta\hat{\ }f \preccurlyeq \sigma'$. As $\beta\hat{\ }f$ was visited at stage s_0, there is at least one marker, namely $\lambda(x, m, s_0)$ where $x = x(\beta)$ and $m = m(\beta)$, that has not been enumerated into A

by stage s_0. The location marker $p(\sigma')$ was chosen to be large at the substage when σ' was visited, and so is larger than $\lambda(x, m, s_0)$. If at some later stage t we enumerate $\lambda(x, m, s_0)$ into A, then A changes below $p(\sigma')$. If we are to have $\Gamma_e^{\Delta_e^A} = A$, then $\Delta_e^A[t] \upharpoonright \gamma_e(p(\sigma'))[s_0]$ is incomparable with $\Delta_e^A[s_0] \upharpoonright \gamma_e(p(\sigma'))[s_0]$. Therefore $m_\tau(\Delta_e^A[t] \upharpoonright d_\tau[t]) = k - 1$, and so m_τ has not decreased in capital along Δ_e^A. If we later arrive at β, we will play $\beta\hat{}\infty$ and initialize σ' as it is to the right of $\beta\hat{}\infty$. If we visit σ' again, a new location marker will be chosen, which must be larger than at least one marker of any node β such that $\tau < \beta\hat{}f \leqslant \sigma'$.

Let s_0 be the least genuine τ-stage. As τ is genuinely visited at stage s_0 there can be no link (τ', σ') at stage s_0 with $\tau' < \tau < \sigma'$. Fix $k \in \omega$ and suppose for contradiction that $\sigma <$ TP devoted to $\mathcal{N}_{e,k}$ is never genuinely visited. Then there is some permanent link (τ'', σ'') with $\tau < \tau'' < \sigma < \sigma''$. Suppose the link (τ'', σ'') was placed at stage s_1. Then as σ'' is genuinely visited at stage s_1 and $\sigma < \sigma''$, σ is genuinely visited at stage s_1. Contradiction. At stage s_1 we define the location marker $p(\sigma)$ and create a link (τ, σ). Let s_2 be the stage at which we travel the link to σ and define m_τ to win \$$k$ on $\Delta_e^A[s_2]$. If we visit a node that is below τ but to the left of σ then, as in the previous paragraph, m_τ will then have capital $k - 1$. However $\sigma <$ TP, and so this will occur only finitely many times. Let s_3 be the greatest stage at which σ is initialized. We remove any link over σ as part of the initalization. At the next σ-stage after s_3 we will genuinely visit σ and place a link if we see that $m_\tau(\Delta_e^A[s_3] \upharpoonright d_\tau[s_3]) < k$. Let s' be the stage at which we travel the link to σ and define more of m_τ. As we never visit a node to the left of σ, no marker belonging to a node β such that $\tau \prec \beta\hat{\ } f \preccurlyeq \sigma$ will be enumerated after stage s'. As in the previous paragraph, A cannot change below the use $\delta_e(\gamma_e(l(\tau))[s']$ and $m_\tau(\Delta_e^A) \geqslant k$. □

Lemma 2.5.4. *For every $x \in \omega$, $\mathcal{H}_x$ is satisfied.*

Proof. Suppose $x \notin$ Cof. Let $\beta \prec$ TP be the node devoted to $\mathcal{H}_x$ with $m(\beta)$ least such that $\beta_{x,m}$ is not permanently linked over by τ's of stronger priority for all $m \geqslant m(\beta)$. Let $m_0 = m(\beta)$. We show that $\lim_t \Lambda^A(x, m, t) = 0$ for all $m \geqslant m_0$. We will have $\Lambda^A(x, m, t) = 0$ unless the marker $\lambda(x, m, t)$ is enumerated into A. Thus we must show that we eventually stop enumerating the markers $\lambda(x, m, t)$ into A. As $[m_0, \infty) \nsubseteq W_x$, the Π_2 test which measures whether $[m, \infty) \subseteq W_x$ will eventually always say "no". So eventually the $\beta_{x,m}$ nodes will stop putting their markers into A and redefining $\Lambda^A(x, m, t)$. Therefore the only way we will enumerate the markers is if there is a link (τ, σ) over $\beta_{x,m}$ with $\tau \prec \beta_{x,m}\hat{\ }\infty \preccurlyeq \sigma$ and τ is accessible. We must show that if there is such a link then τ is accessible at only finitely many stages.

For τ with $e(\tau) > x$, we will restart τ below $\beta_{x,m}\hat{\ }\infty$. If $\beta\hat{\ }\infty \preccurlyeq \tau \prec \beta_{x,m_1} \preccurlyeq \sigma$ and a link (τ, σ) is placed over β_{x,m_1} for some $m_1 > m$, then as $\beta\hat{\ }\infty$ can be visited at most finitely many times, only finitely many markers $\lambda(x, m_1, t)$ will be enumerated.

Now suppose τ is above β and we place a link (τ, σ) over β. Subcase (2a) of the construction will enumerate markers $\lambda(x, m, t)$ only if $\beta_{x,m}\hat{\ }\infty \leqslant \sigma$. As σ must be below $\beta\hat{\ }f$, the markers $\lambda(x, m, t)$ will not be enumerated, and we will have $\lim_t \Lambda^A(x, m, t) = 0$.

Finally, if $\beta'_{x,m}$ is another node on another path of the priority tree which is visited infinitely often, we must ensure that it does not enumerate all of the markers $\lambda(x, m, t)$. The Π_2 test performed at $\beta'_{x,m}$ is the same test which is performed at $\beta_{x,m}$ and so will eventually always say "no". Therefore the marker $\lambda(x, m, t)$ will only be enumerated if there is a link (τ, σ) over $\beta'_{x,m}$ with $\beta'_{x,m}\hat{\ }\infty \leqslant \sigma$. As $\beta'_{x,m}$ is to the right of TP it will be initialized infinitely many times. Any link over $\beta'_{x,m}$ will be removed as part of the initialization, and so $\beta'_{x,m}$ is not permanently linked over. The marker $\lambda(x, m, t)$ is enumerated capriciously only if $\beta'_{x,m}\hat{\ }\infty \leqslant \sigma$. For the marker to be enumerated infinitely many times we must visit σ below $\beta'_{x,m}\hat{\ }\infty$ and place a link back to τ. However if $\beta'_{x,m}\hat{\ }\infty$ is visited infinitely many times, this contradicts $x \notin \mathrm{Cof}$. Therefore $\beta'_{x,m}$ cannot enumerate infinitely many of its markers.

Now suppose that $x \in \mathrm{Cof}$ and $[m_0, \infty) \subseteq W_x$. Let $\beta \prec \mathrm{TP}$ be the node devoted to $\mathcal{H}_x$ with $m(\beta)$ least such that $\beta_{x,m}$ is not permanently linked over by τ's of stronger priority for all $m \geqslant m(\beta)$. Let $m' = \max\{m(\beta), m_0\}$. We show that $\lim_t \Lambda^A(x, m, t) = 1$ for all $m \geqslant m'$.

We first show that $\lim_s c_\tau(x, m)[s]$ exists for all $\tau \leqslant \beta$ and so all markers $\lambda(x, m, q) > \max_{\tau\leqslant\beta} \lim_s c_\tau(x, m)[s]$ may be enumerated into A. The value of $c_\tau(x, m)$ can be increased only when the mother node τ links to some worker σ with $\sigma \prec \beta'_{x,m}$ and subsequently defines more of the martingale m_τ. The priority tree is finite-branching, and so there are only finitely many nodes $\beta_{x,m}$. Consider

$$\langle e', k' \rangle = \max \cup_{\beta_{x,m}} \{\langle e, k \rangle \mid \sigma \prec \beta_{x,m} \text{ has mother } \tau \text{ and is devoted to } \mathcal{N}_{e,k}\}.$$

Let σ_0 be the node on TP devoted to $\mathcal{N}_{e',k'}$, and suppose that no node to the left of σ_0 is visited after stage s_0. When we genuinely visit any node $\sigma \preccurlyeq \sigma_0$ with mother node τ, the construction will check to see whether $m_\tau(\Delta^A_e[s] \upharpoonright d_\tau[s]) \geqslant k(\sigma)$. If so, we will play the d outcome. As $\sigma \prec$ TP, the martingale will never decrease in capital and $c_\tau(x, m)$ cannot be increased again after travelling a link to some σ' devoted to $\mathcal{N}_{e,k(\sigma)}$. If $m_\tau(\Delta^A_e \upharpoonright d_\tau[s]) < k(\sigma)$, then we will create a link from σ back to τ at stage $s > s_0$. If the link is permanent then $c_\tau(x, m)$ will never be increased. If we later define more of m_τ we will then increase $c_\tau(x, m)$. Again, as $\sigma \prec$ TP, the martingale will never decrease in capital and $c_\tau(x, m)$ cannot be increased again after travelling a link to some σ' devoted to $\mathcal{N}_{e,k(\sigma)}$. There are only finitely many worker nodes $\sigma \preccurlyeq \sigma_0$ with mother node τ. Therefore $\lim_s c_\tau(x, m)[s]$ exists for all $\tau \preccurlyeq \beta$.

We will restart all τ nodes with $e(\tau) > x$ below $\beta_{x,m'} \hat{\ } \infty$. As $[m', \infty) \subseteq W_x$, the Π_2 test which measures whether $[m, \infty) \subseteq W_x$ will say "yes" infinitely many times for all $m \geqslant m'$. If a link (τ, σ) is placed over $\beta_{x,m'}$, and so $\beta_{x,m'} \hat{\ } f \preccurlyeq \sigma$, we will perform a scouting report when we arrive at τ. Suppose the link (τ, σ) is placed at stage s_0 and that s_1 is the least stage greater than s_0 at which the Π_2 test says "yes". At the least τ-stage after s_1, the scouting report will be successful, and $\beta_{x,m'} \hat{\ } \infty$ will be accessible. We then will remove the link (τ, σ) and enumerate the marker $\lambda(x, m', t)$. In this way all of

$\beta_{x,m'}$'s markers will eventually be enumerated.

Now suppose that $\beta_{x,m'}\,\hat{}\,\infty \preccurlyeq \tau \prec \beta_{x,m_1} \prec \sigma$. The node β_{x,m_1} has only the ∞ outcome, as it is below $\beta_{x,m'}\,\hat{}\,\infty$ with $m' < m_1$. If β_{x,m_1} is on GTP then it will enumerate its markers whenever it is accessible and define Λ such that $\lim_t \Lambda^A(x, m_1, t) = 1$. If β_{x,m_1} is not on GTP then the link (τ, σ) must be permanent. As in the previous paragraph, $\beta_{x,m'}\,\hat{}\,\infty$ will be accessible at infinitely many stages. When we arrive at τ and see the link, we will enumerate markers of the form $\lambda(x, m_1, q)$. Therefore all of β_{x,m_1}'s markers will eventually be enumerated and $\lim_t \Lambda^A(x, m_1, t) = 1$. $\square$

This concludes the verification of the construction, and the proof of Theorem 2.1.8.

Chapter 3

DNR and incomparable Turing degrees

This chapter is joint work with Mingzhong Cai and Noam Greenberg, and has appeared in [10].

3.1 Introduction

One way in which we might consider a Turing degree to be computationally strong is if it computes a *diagonally noncomputable function*: a function $f\colon \omega \to \omega$ such that $f(e) \neq \varphi_e(e)$ when the latter is defined. Then f is thought to be far from computable because it can give $f(e)$ as a counterexample to the assertion that $f = \varphi_e$.

We might wonder whether a Turing degree can be simultaneously strong in this sense, but weak in another. Sacks in [39] asked whether there exists a degree which both computes a diagonally noncomputable (DNC) function and is minimal. Kumabe ([32]) answered this affirmatively using a technique known as forcing with "bushy trees". Since then the technique has been used in several results, including some in algorithmic randomness and reverse mathematics. See [29] for a survey of such results.

In this chapter we extend this technique to show:

Theorem 3.1.1. *There is an initial segment $\boldsymbol{a}_1 < \boldsymbol{a}_2 < \boldsymbol{a}_3 < \cdots$ of the Turing degrees such that each $\boldsymbol{a}_{n+1}$ is a* DNC *degree relative to $\boldsymbol{a}_n$.*

The theorem has a corollary in reverse mathematics.

Corollary 3.1.2. *The system DNR$_0$ does not imply Turing incomparability, in fact it does not imply the existence of a pair of Turing incomparable reals.*

Although no knowledge of reverse mathematics is required for the proof of the theorem, we provide some background for motivation.

Reverse mathematics is a programme in the foundations of mathematics with deep connections with computability theory. We consider subsystems of second order arithmetic. Our models M consist of two parts. The first part is a set $|M|$ which we interpret as our *numbers*, and the second part is a collection $\mathcal{S}_M$ of subsets of $|M|$ which we interpret as our *sets*. We are often interested in so-called ω-models, where $|M|$ is simply ω, the standard natural numbers. Our collection of sets is usually a Turing ideal. That is, it is closed under computable join and downward closed under Turing reducibility.

We consider axiom systems and models of such systems. One important system is known as WKL_0 (weak König's lemma). It ensures that our Turing ideal contains a completion of Peano Arithmetic. In [31], Kučera and Slaman solved a long-standing open problem by showing that for every model M of WKL_0, if $x \in \mathcal{S}_M$ is noncomputable, then there is some $y \in \mathcal{S}_M$ which is Turing incomparable with x. We say that Turing incomparability holds in every ω-model of WKL_0. This was improved by Conidis [15] to show that Turing incomparability holds in ω-models of the weaker system $WWKL_0$, the system which ensures the existence of a Martin-Löf random set.

A prominent system below $WWKL_0$ is DNR_0, the system which ensures the existence of a DNC function. The systems $WWKL_0$ and DNR_0 were first separated by Ambos-Spies et al. [1] using a tame version of bushy tree forcing. Our corollary shows that Turing incomparability does *not* necessarily hold in models of the weaker system DNR_0.

We prove Theorem 3.1.1 in four steps. The third step (in Section 3.4) provides the construction, for each $n < \omega$, of an initial segment $\boldsymbol{a}_1 < \cdots < \boldsymbol{a}_n$ of the desired infinite sequence $\langle \boldsymbol{a}_k \rangle$. The fourth and last step (in Section 3.5) shows how to string these constructions together and so prove Theorem 3.1.1. The first two steps serve as an introduction to the construction of Section 3.4. In Section 3.2 we recast Kumabe's construction in the language of forcing that we subsequently use. In Section 3.3 we discuss the case $n = 2$ (the construction of a minimal DNC degree $\boldsymbol{a}_1$ and a strong minimal cover $\boldsymbol{a}_2$ of $\boldsymbol{a}_1$ which is DNC relative to $\boldsymbol{a}_1$). Recall that $\boldsymbol{b}$ is a strong minimal cover of $\boldsymbol{a}$ if $\boldsymbol{b} > \boldsymbol{a}$, but for all $\boldsymbol{c} < \boldsymbol{b}$ we have $\boldsymbol{c} \leqslant \boldsymbol{a}$. We say that the function f is DNC relative to $\boldsymbol{a}$ if for some function a of degree $\boldsymbol{a}$, $f(e) \neq \Phi_e^a(e)$ where the latter is defined, and we write $f \in \mathrm{DNC}^a$. We say that the degree $\boldsymbol{b}$ is DNC relative to $\boldsymbol{a}$ if it contains some function which is DNC relative to $\boldsymbol{a}$.

3.1.1 Fast-growing functions

Below we use trees (or tree systems) which are fairly "bushy" but associated with them we will have sets of "bad" strings which we want to avoid. In the first step we use infinite trees and for example declare every string which is not DNC to be bad. We then extend the bad set of strings when we force divergence or force a functional to be constant on a tree. We cannot simply remove the bad strings from the tree because the trees will be computable whereas the set of bad strings will be c.e. To ensure that most strings are not bad, and that the construction can proceed, we will require that the tree is h-bushy and that the bad set of strings is b-small above the stem of the tree, where h grows much more quickly than the order-function b. Here we discuss the notion of relative quickness that we will use.

For an equivalence notion of rate of growth we close under relative elementary recursive functions. (We could use relative primitive recursive functions but this is not needed.) For any order function h one defines the class of order functions which are obtained from h using a list of rules such as substitution and bounded summation and multiplication.

We are only concerned with rates of growth. If h grows sufficiently quickly then g is bounded by a function elementary in h if and only if it is dominated by an iterated composition of h with itself. In particular, the elementary recursive functions are those which are bounded by iterated exponentials.

It will be convenient to consider functions that may be undefined on a finite initial segment of ω.

Definition 3.1.3. Let Q denote the collection of nondecreasing computable functions $h\colon \omega \to [2, \omega)$ satisfying $h(n) \geqslant 2^n$ for all n.

For $h \in Q$ let $h^{(1)} = h$ and for $k \geqslant 1$, $h^{(k+1)} = h \circ h^{(k)}$. For two functions h and g in Q we say that h *majorises* g if $h(n) \geqslant g(n)$ for all n (and write $h \geqslant g$). We say that $h \geqslant g$ *above* m if $h(n) \geqslant g(n)$ for all $n \geqslant m$. We say that h *dominates* g if $h \geqslant g$ above some m (and write $h \geqslant^* g$).

We will use the fact that iterated exponentials of h are dominated by iterates of h. For example:

Example 3.1.4. Let $h \in Q$. Let $g(n) = \prod_{m \in [0,n)} h(m)$. Then $g \leqslant^* h^{(3)}$. For $g \leqslant h^h$ whereas $h^{(2)} \geqslant 2^h$ and $h^{(3)} \geqslant 2^{2^h}$, and $2^{2^h} \geqslant^* h^h$.

Definition 3.1.5. Let $h, g \in Q$. We say that *h dominates the iterates of g uniformly*, and write $h \gg g$, if there is a computable sequence $\langle d_k \rangle$ such that for all $k \geqslant 1$, $h \geqslant g^{(k)}$ on the interval (d_k, ω).

The relation $\gg$ on Q is transitive. Indeed if $h \gg g$, $h' \geqslant^* h$ and

$g \geqslant^* g'$ then $h' \gg g'$. Further, $h \gg g^{(k)}$ for all k, and so for example $h \gg 2^g$.

The following density lemma will be used to keep extending conditions.

Lemma 3.1.6. *For all $h, g \in Q$ such that $h \gg g$ there is some $f \in Q$ such that $h \gg f \gg g$.*

Proof. The idea is to gradually let f copy $g^{(k)}$. If f is bounded by $g^{(k)}$ for a long time, then for a shorter time we can ensure that $f^{(k)}$ is bounded by $g^{(k^2)}$, so we do this until the point where h starts to majorise $g^{((k+1)^2)}$, and only then start copying $g^{(k+1)}$.

Since g is nondecreasing and dominates the identity, each $g^{(k)}$ is nondecreasing and $g^{(k+1)} \geqslant g^{(k)}$.

Let $k \geqslant 1$, $e \geqslant 0$ and let f be a function. Suppose that $f \leqslant g^{(k)}$ on the interval $[0, g^{(k^2)}(e)]$ (actually the interval $[0, g^{(k^2-k)}(e)]$ will suffice). Then $f^{(k)} \leqslant g^{(k^2)}$ on the interval $[0, e]$: by induction on $j \leqslant k$ we see that $f^{(j)} \leqslant g^{(kj)}$ on the interval $[0, g^{(k(k-j))}(e)]$.

Let $\langle d_k \rangle$ witness that $h \gg g$. We may assume that $\langle d_k \rangle$ is nondecreasing.

We define a computable sequence $-1 = a_0 \leqslant a_1 \leqslant \cdots$ and then define f by letting $f = g^{(k)}$ on the interval $(a_{k-1}, a_k]$. So the sequence $\langle a_{k-1} \rangle$ witnesses that $f \gg g$. But also $f \leqslant g^{(k)}$ on the interval $[0, a_k]$ for all $k \geqslant 1$. So we let $a_k = g^{(k^2)}(d_{(k+1)^2})$. This ensures that $f^{(k)} \leqslant g^{(k^2)}$ on $[0, d_{(k+1)^2}]$, which in turn shows that $h \geqslant f^{(k)}$ on the interval $(d_{k^2}, d_{(k+1)^2}]$. Since $f \in Q$, $f^{(m)} \geqslant f^{(k)}$ if $m \geqslant k$, so the sequence $\langle d_{k^2} \rangle$ witnesses that $h \gg f$. □

3.1.2 Other notation and conventions

A *string* is a finite sequence of natural numbers, an element of $\omega^{<\omega}$. If σ is a string then we let $\sigma^{\preccurlyeq}$ be the collection of strings which extend σ, and $[\sigma]^{\prec}$ be the set of elements of Baire space ω^{ω} which extend σ. If C is a set of strings then $C^{\preccurlyeq} = \bigcup_{\sigma \in C} \sigma^{\preccurlyeq}$ and so $[C]^{\prec} = \bigcup_{\sigma \in C} [\sigma]^{\prec}$.

We may assume that for any Turing functional Γ and for any string τ, the domain of $\Gamma(\tau)$ is downwards closed. Thus Γ determines a monotone computable map $\tau \mapsto \Gamma(\tau)$ from strings to strings, which induces a partial computable function on Baire space: $\Gamma(x) = \bigcup \{\Gamma(\tau) : \tau \prec x\}$.

We let lowercase Greek letters denote strings, lowercase Roman letters denote elements of Baire space, and uppercase Roman letters denote sets of strings and sometimes subsets of Baire space.

3.1.3 Compactness, splittings and computability

Definition 3.1.7. A subset X of Baire space is *computably bounded* if some computable function majorises every element of X.

Every computably bounded and closed subset of Baire space is compact.

The following is well-known.

Lemma 3.1.8. *Let $X \subseteq \omega^{\omega}$ be Π^0_1 and computably bounded; let*

$f\colon X \to 2^\omega$ be a computable function.[1]

- *If f is constant on X then this constant value is computable.*
- *If f is 1-1 on X then for all $x \in X$, $x \equiv_{\mathrm{T}} f(x)$.*

Proof. Suppose that f is constant on X; let $f[X] = \{y\}$. The fact that X is computably bounded implies that the set of $\alpha \in 2^{<\omega}$ such that $X = f^{-1}\left[[\alpha]^{<}\right]$ is c.e.; this is the set of initial segments of y, so y is computable.

Suppose that f is 1-1 on X. Let $Y = f[X]$. Then Y a Π^0_1 subset of 2^ω and f is a homeomorphism between X and Y. And f^{-1} is computable: the set of pairs (α, τ) such that $[\alpha]^{<} \cap Y \subseteq f\left[[\tau]^{<}\right]$ is c.e. □

If $X \subseteq (\omega^\omega)^2$ and $x \in \omega^\omega$ we let $X_x = \{y : (x, y) \in X\}$.

Lemma 3.1.9. *Let $X \subseteq (\omega^\omega)^2$ be Π^0_1 and computably bounded. Let $f\colon X \to 2^\omega$ be computable and suppose that the collection of sets $f[X_x]$ for $x \in$ dom X are pairwise disjoint. Then for all $(x, y) \in X$, $x \leqslant_{\mathrm{T}} f(x, y)$.*

Proof. For $\tau \in \omega^{<\omega}$ let $X_\tau = \bigcup_{x \in [\tau]^{<}} X_x$. The set of pairs (τ, C) where $C \subseteq 2^\omega$ is clopen and $f[X_\tau] = C \cap f[X]$ is c.e. □

[1] Here we think of X and 2^ω as computable metric spaces. A computable function from X to 2^ω is given by a uniform Turing reduction.

3.1.4 Forcing with closed sets

We quickly remind the reader of just a few definitions needed for our development of forcing. A good reference for forcing more generally in computability theory is Chapter 3 of [42].

A notion of forcing is a partial order $(\mathbb{P}, \leqslant)$. We call the elements of $\mathbb{P}$ *conditions*, and if $q \leqslant p$, then we say that q *extends* p, or that q is *below* p. A subset F of $\mathbb{P}$ is called a *filter* if is is upwards closed, and every pair of elements in F has a common extension in F. A subset D of $\mathbb{P}$ is *dense* if every condition $p \in \mathbb{P}$ has an extension in D.

Definition 3.1.10. Let $\mathbb{P}$ be a notion of forcing. Suppose that with each condition $p \in \mathbb{P}$ we associate a closed subset X^p of Baire space. We call this assignment *acceptable* if:

(a) for all $p \in \mathbb{P}$, X^p is nonempty;

(b) if q extends p then $X^q \subseteq X^p$; and

(c) for every m, the set of conditions $p \in \mathbb{P}$ such that $X^p \subseteq [\sigma]^{\prec}$ for some string σ of length m is dense in $\mathbb{P}$.

(Below we will consider finite powers $(\omega^\omega)^n$ of Baire space, but these are of course effectively isomorphic to Baire space.)

Recall the *Borel codes* for Borel subsets of Baire space. These can be identified with propositional sentences in $L_{\omega_1,\omega}$. To be precise:

- Every finite set of strings C is a Borel code;

- If C is a Borel code then $\neg C$ is a Borel code;
- If $\mathcal{C}$ is a countable set of Borel codes, then $\bigvee \mathcal{C}$ and $\bigwedge \mathcal{C}$ are Borel codes.

The semantics are obvious (a finite set of strings C defines the set $[C]^{<}$); if C is a Borel code then we let $\lfloor C \rfloor$ be the Borel subset of Baire space defined by C.

Suppose that $\mathbb{P}$ is a notion of forcing equipped with an acceptable assignment of closed sets X^p. We define the forcing relation $p \Vdash C$ between conditions in $\mathbb{P}$ and Borel codes C. We start with strong forcing.

Definition 3.1.11. Let C be a Borel code and let $p \in \mathbb{P}$. We say that *p strongly forces C* if $X^p \subset \lfloor C \rfloor$. We write $p \Vdash^* C$.

Now by recursion on Borel codes C we define forcing.

- For a finite set of strings D, $p \Vdash D$ if the collection of conditions which strongly force D is dense below p.
- $p \Vdash \neg C$ if no extension of p forces C.
- $p \Vdash \bigwedge \mathcal{C}$ if $p \Vdash C$ for all $C \in \mathcal{C}$.
- $p \Vdash \bigvee \mathcal{C}$ if the set of conditions which force some element of $\mathcal{C}$ is dense below p.

The basic properties of forcing hold.

Lemma 3.1.12. *Let $p \in \mathbb{P}$ and let C be a Borel code.*

1. *No condition forces both C and $\neg C$.*
2. *The set of conditions which decide C is dense in $\mathbb{P}$.*
3. *If q extends p and $p \Vdash C$ then $q \Vdash C$.*
4. *If the set of conditions which force C is dense below p then $p \Vdash C$.*

Forcing equals truth. That is, every sentence in our language is true if and only if it can be forced. It will be convenient to consider directed subsets of $\mathbb{P}$ rather than filters; of course the upwards closure of a directed set is a filter, so we can always pass to filters without adding information. Genericity for directed sets is defined using dense *open* sets: dense subsets of $\mathbb{P}$ which are closed downwards (closed under taking extensions). Note that the dense sets of conditions mentioned above are all open.

Suppose that $G \subset \mathbb{P}$ is a directed set. If G meets all of the dense open sets of conditions guaranteed by (c) above, then $\bigcap_{p\in\mathbb{P}} X^p$ is a singleton that we denote by $\{x^G\}$. (This uses the completeness of Baire space; we do not need the sets X^p to be compact.)

In the rest of the chapter, the statement "for all sufficiently generic $G \subset \mathbb{P}$ ..." means: there is a countable collection $\mathcal{D}$ of dense open subsets of $\mathbb{P}$ such that for every directed subset of $\mathbb{P}$ meeting all the sets in $\mathcal{D}$, ...

Lemma 3.1.13. *Let C be a Borel code. If $G \subset \mathbb{P}$ is a sufficiently generic directed set then $x^G \in \lfloor C \rfloor$ if and only if $p \Vdash C$ for some $p \in G$.*

Proof. First note that if $p \in G$ and $p \Vdash^* C$ then $x^G \in \lfloor C \rfloor$. On the other hand, suppose that D is a finite set of strings, and suppose that $x^G \in [D]^{\prec}$: there is some $\tau \prec x^G$ such that $\tau \in D$. By assumption, there is some string η of length $|\tau|$ and some $p \in G$ such that $X^p \subseteq [\eta]^{\prec}$. Then $\eta = \tau$, and so $p \Vdash^* D$, which implies that $p \Vdash D$.

The rest of the argument follows the usual proof of the equivalence of forcing and truth for generic filters. □

Since every condition can be extended to a sufficiently generic directed set, we conclude:

Corollary 3.1.14. *Let $p \in \mathbb{P}$ and let C be a Borel code.*

1. *$p \Vdash C$ if and only if for every sufficiently generic directed set G, if $p \in G$ then $x^G \in \lfloor C \rfloor$.*
2. *If $\lfloor C \rfloor \subseteq \lfloor C' \rfloor$ and $p \Vdash C$ then $p \Vdash C'$.*
3. *If $p \Vdash^* C$ then $p \Vdash C$.*

In light of (2) we write $p \Vdash x^G \in A$ when A is a Borel subset of Baire space, rather than a code for such a set.

3.1.5 Simplified iterated forcing

We give a not-completely-standard definition for restriction maps between notions of forcing.

Definition 3.1.15. Let $\mathbb{P}$ and $\mathbb{Q}$ be partial orderings. A *restriction map* from $\mathbb{Q}$ to $\mathbb{P}$ is an order-preserving map i from $\mathbb{Q}$ to $\mathbb{P}$ such that for all $q \in \mathbb{Q}$, the image of $\mathbb{Q}(\leqslant q)$ (the set of extensions of q) under i is dense below $i(q)$.

That is, for all $q \in \mathbb{Q}$ and $p \leqslant i(q)$ there is some $r \leqslant q$ in $\mathbb{Q}$ such that $i(r) \leqslant p$.

Lemma 3.1.16. *Let $i\colon \mathbb{Q} \to \mathbb{P}$ be a restriction map.*

1. *If $G \subset \mathbb{Q}$ is a directed set then $i[G] \subset \mathbb{P}$ is a directed set.*
2. *If $D \subseteq \mathbb{P}$ is dense and open then $i^{-1}[D] \subseteq \mathbb{Q}$ is dense and open.*

Hence for any collection $\mathcal{D}$ of dense open subsets of $\mathbb{P}$ there is a collection $\mathcal{E}$ of dense open subsets of $\mathbb{Q}$ such that if $G \subset \mathbb{Q}$ is a directed set which meets every set in $\mathcal{E}$, then $i[G]$ is a directed set which meets every set in $\mathcal{D}$. In other words, if G is sufficiently generic then so is $i[G]$.

Suppose that $\mathbb{P}$ and $\mathbb{Q}$ have acceptable assignments of closed sets $X^p \subseteq \omega^\omega$ for $p \in \mathbb{P}$ and $Y^q \subseteq (\omega^\omega)^2$ for $q \in \mathbb{Q}$. Suppose that $i\colon \mathbb{Q} \to \mathbb{P}$ is a restriction map and further that for all $q \in \mathbb{Q}$, $X^{i(q)} \supseteq \operatorname{dom} Y^q$. Let $G \subset \mathbb{Q}$ be sufficiently generic; we denote the generic pair of reals by (x^G, y^G). Then $x^{i[G]} = x^G$.

3.1.6 The plan

To prove Theorem 3.1.1, for each $n < \omega$ we define a notion of forcing $\mathbb{P}_n$ which adds an initial segment of the degrees of length n, each degree DNC relative to the one below it. We then show that there are restriction maps from each $\mathbb{P}_n$ to $\mathbb{P}_{n-1}$. This will allow us to obtain generic $G_n \subset \mathbb{P}_n$ which are coherent, from which we will obtain the desired ω-sequence of degrees.

3.2 A DNC minimal degree

Khan (see [29]) showed that for any Turing degree $\boldsymbol{x}$ there is a function which is DNC relative to $\boldsymbol{x}$ and of minimal Turing degree. He presented an elaboration on the Kumabe-Lewis construction using the language of forcing in computability (rather than give an explicit construction). The extra complication is due to the fact that the set of strings which are not $\mathrm{DNC}^{\boldsymbol{x}}$ is c.e. in $\boldsymbol{x}$, rather than merely c.e. We have no access to this set when defining the computable trees. For this reason Khan needs to use trees with terminal elements (and the set of terminal elements is co-c.e. but not computable).

In this section we present a proof of the original Kumabe-Lewis theorem using the language of forcing. We use c.e. sets of bad strings and trees with no terminal elements.

3.2.1 Trees and forests

We follow [1, 23, 29] and use trees which are sets of strings rather than function trees (as in [9, 32]). We localise to basic clopen sets.

Recall that for a string σ, $\sigma^{\preccurlyeq}$ is the set of strings extending σ. A *tree above* σ is a nonempty subset of $\sigma^{\preccurlyeq}$ which is closed in $\sigma^{\preccurlyeq}$

under taking initial segments. A set of strings A is prefix-free if no two distinct elements of A are comparable. If A is a finite prefix-free set of strings then a *forest above* A is a set $T \subseteq A^{\preccurlyeq}$ such that for all $\sigma \in A$, $T \cap \sigma^{\preccurlyeq}$ is a tree above σ. In particular we require that $A \subseteq T$. When we just say “tree” we mean a tree above σ for some σ; the string σ will usually be clear from the context or unimportant. The same holds for forests. We will mostly only use finite forests, but will use both finite and infinite trees.

Let T be a forest and let $\tau \in T$. An *immediate successor* of τ on T is a string $\tau' \succ \tau$ on T such that $|\tau'| = |\tau| + 1$. A *leaf* of a forest T, also known as a *terminal* element of T, is a string on T which has no proper successors on T.

A *subtree* of a tree T is a subset $S \subseteq T$ which is a tree. Note that the stem of S may equal the stem of T, or properly extend the stem of T. If T is a tree and $\tau \in T$ then the full subtree of T above τ is $T \cap \tau^{\preccurlyeq}$, the set of strings on T which extend τ.

If T is a tree above σ then $[T]$ is the set of infinite paths of T, the set of $x \in \omega^{\omega}$ such that $x\restriction_n \in T$ for all $n \geqslant |\sigma|$. This is a closed subset of ω^{ω}. Recall that $[\sigma]^{\prec}$ is the set of extensions of σ in Baire space; in our notation, $[\sigma]^{\prec} = [\sigma^{\preccurlyeq}]$.

A tree T is *bounded* by a function h if for all $\tau \in T$, $\tau(n) < h(n)$ for all $n \leqslant |\tau|$. It is *computably bounded* if h can be taken to be computable. If T is computably bounded then so is $[T]$ (Definition 3.1.7).

3.2.2 Bushy notions of largeness

The basic notions of "bushiness" were extended from constant bounds to order functions, see [8, 29]. We recall the definitions and basic properties. A *bounding function* is a computable function from ω to $[2, \omega)$.

Definition 3.2.1. Let T be a forest above a finite prefix-free set of strings A; let h be a bounding function. We say that T is *h-bushy* if every nonterminal $\tau \in T$ has at least $h(|\tau|)$ many immediate successors on T.

Definition 3.2.2. Let A be a finite prefix-free set of strings and let B be a set of strings. Let h be a bounding function. The set B is *h-big above* A if there is a finite forest T above A which is h-bushy, all of whose leaves are elements of B.

If A is an infinite set of strings then we say that B is h-big above A if B is h-big above every finite, prefix-free subset of A.

If B is not h-big above A then we say it is *h-small* above A.

If A is a singleton $\{\sigma\}$ then we say that B is h-big (or h-small) above σ. A set B is h-big above A if and only if the set of minimal strings in B is h-big above A. We thus often use the notion for either prefix-free sets of strings, or for *open* sets of strings – those that are upwards closed (closed under taking extensions). Also note that sometimes we do not assume that B only contains extensions of A, but of course for this notion it suffices to look at $B \cap A^{\preccurlyeq}$.

The following remark is trivial. Its generalisations in later sections will be less so.

Remark 3.2.3. Suppose that B is a set of strings, h-big above A, and that $A, B \subseteq T$ for some tree T. Then any forest S which witnesses that B is h-big above A is a subset of T.

The basic combinatorial properties of this notion of largeness have been repeatedly observed [27, 32, 23, 29].

Lemma 3.2.4 (Big subset property)**.** *Let h and g be bounding functions. Let B and C be sets of strings, let σ be a string, and suppose that $B \cup C$ is $(h+g)$-big above σ. Then either B is h-big above σ or C is g-big above σ.*

Here it is important that we work above a single string σ and not above any finite A.

Proof. Let T be a tree which witnesses that $B \cup C$ is $(h+g)$-big above σ. Label a leaf τ of T "B" if it is in B, and "C" otherwise. Now if $\rho \in T$ and all immediate successors of ρ have been labelled then since ρ has at least $h(|\rho|) + g(|\rho|)$ immediate successors on T, either at least $h(|\tau|)$ of these are labelled "B" or at least $g(|\tau|)$ of them are labelled "C". In the first case label ρ "B", in the other label it "C". Eventually σ is labelled. If σ is labelled "B" then the set of $\rho \in T$ labelled "B" form a tree which witnesses that B is h-big above σ; and similarly if σ is labelled "C". □

Lemma 3.2.5 (Concatenation property)**.** *Let h be a bounding function. Let A, B and C be sets of strings. Suppose that B is h-big* *above A, and that C is h-big above every $\tau \in B$. Then C is h-big above A.*

Proof. Let A' be a finite, prefix-free subset of A. Let T be a forest which witnesses that B is h-big above A'. For a leaf τ of T let R_τ be a tree which witnesses that C is h-big above τ. Then $T \cup \bigcup R_\tau$, where τ ranges over the leaves of T, witnesses that C is h-big above A'. □

The concatenation property will sometimes be used to recursively build bushy trees meeting infinitely many big sets. Again the following are fairly immediate; their generalisations in the next sections will be less so.

Definition 3.2.6. A forest R is an *end-extension* of a forest S if every string in $R \backslash S$ extends some leaf of S.

(This is not the same as the usual definition for partial orderings, but under the usual definition, any tree extension is an end-extension.) The argument proving the concatenation is broken up to show:

Lemma 3.2.7. *Let A, B, C be sets of strings, and let h be a bounding function.*

1. *Suppose that C is h-big above every $\tau \in B$. Then C is h-big above B.*
2. *Suppose that A is prefix-free and finite; suppose that B is h-big above A and that C is h-big above B. Then any forest which*

witnesses that B is h-big above A has an end-extension which witnesses that C is h-big above A.

Remark 3.2.8. Throughout, we will assume that whenever we are given a set of strings which is guaranteed to have some largeness property, then this set is the set of leaves of a forest witnessing this property. For example, suppose that we are given a set B which is h-big above some σ. We will assume, often without mentioning it, that B is finite, that it is prefix-free, and that every string in B extends σ.

3.2.3 The notion of forcing and the generic

Let B_{DNC} be the set of strings σ that are not initial segments of diagonally noncomputable functions: $\sigma(e) = J(e)\downarrow$ for some $e < |\sigma|$, where J is a fixed universal jump function, for example $J(e) = \varphi_e(e)$.

Let T be a tree. We say that a set of strings $B \subseteq T$ is *open in T* if it is upwards closed in T: if $\sigma \in B$ and $\tau \succcurlyeq \sigma$ is in T then $\tau \in B$.

We let $\mathbb{P}_1$ be the set of tuples $p = (\sigma^p, T^p, B^p, h^p, b^p)$ satisfying:

1. T^p is a computably bounded, computable tree above σ^p with no leaves.

2. $h^p \in Q$ and T^p is h^p-bushy.

3. $B^p \subset T^p$ is c.e. and open in T^p, and $B^p \supseteq B_{\mathrm{DNC}} \cap T^p$.

4. $b^p \in Q$ and B^p is b^p-small above σ^p.

5. $h^p \gg b^p$ and $h^p \geqslant b^p$ above $|\sigma^p|$.

Lemma 3.2.9. *$\mathbb{P}_1$ is nonempty.*

Proof. The set B_{DNC} is c.e. and is 2-small above the empty string $\langle\rangle$. Fix some $b \in Q$ (and recall that $b \geqslant 2$); and find some $h \in Q$ such that $h \gg b$ and $h \geqslant b$ (for example $h(n) = b^{(n+1)}(n)$). Recall that $h^{<\omega}$ is the set of h-bounded strings. Then $p = (\langle\rangle, h^{<\omega}, B_{\mathrm{DNC}} \cap h^{<\omega}, h, b)$ is a condition in $\mathbb{P}_1$. □

We define a partial ordering on $\mathbb{P}_1$ as follows. A condition q extends a condition p if $\sigma^p \preccurlyeq \sigma^q$, T^q is a subtree of T^p, $B^p \cap T^q \subseteq B^q$, and $h^q \leqslant h^p$ and $b^q \geqslant b^p$ above $|\sigma^q|$.

To use the machinery of forcing developed in Section 3.1.4 we need to associate with each condition $p \in \mathbb{P}_1$ a closed set X^p.

Lemma 3.2.10. *The assignment of closed sets $X^p = [T^p] \backslash [B^p]^{\prec} = [T^p \backslash B^p]$ for $p \in \mathbb{P}_1$ is acceptable (Definition 3.1.10).*

Proof. Requirement (b), that $X^q \subseteq X^p$ if q extends p, follows directly from the definition of the partial ordering on $\mathbb{P}_1$.

Let $p \in \mathbb{P}_1$. Suppose that $[T^p] \subseteq [B^p]^{\prec}$. Since T^p is bounded, $[T^p]$ is compact. This implies that there is a prefix-free, finite set $C \subset B^p$ such that every $\tau \in T^p$ is comparable with some element of C. The collection of strings in T^p extended by some string in C witnesses that B^p is h^p-big above σ^p. Since $h^p \geqslant b^p$ above $|\sigma^p|$ this implies that B^p is b^p-big above σ^p. We get requirement (a): X^p is nonempty.

Again let $p \in \mathbb{P}_1$. Let $m \geqslant |\sigma^p|$. There is some $\tau \in T^p$ of length m above which B^p is b^p-small; otherwise, the concatenation property implies that B^p is b^p-big above σ^p. If B^p is b^p-small above τ then $q = (\tau, T^p \cap \tau^{\preccurlyeq}, B^p \cap \tau^{\preccurlyeq}, h^p, b^p)$ is a condition in $\mathbb{P}_1$ extending p and satisfying $X^q \subseteq [T^q] \subseteq [\tau]^{\prec}$. This gives requirement (c) of Definition 3.1.10. □

As discussed in Section 3.1.4, if $G \subset \mathbb{P}_1$ is sufficiently generic then $\bigcap_{p\in G}[T^p \backslash B^p]$ is a singleton $\{x^G\}$. In fact

$$x^G = \bigcup \{\sigma^p \ : \ p \in G\} .$$

Let $p \in \mathbb{P}_1$; since $B_{\text{DNC}} \cap T^p \subseteq B^p$ we see that $X^p \subseteq \text{DNC}$. Since strong forcing implies forcing (Corollary 3.1.14(3)) we get:

Proposition 3.2.11. *Every condition in $\mathbb{P}_1$ forces that $x^G \in \text{DNC}$.*

Remark 3.2.12. Let A be an open set of strings and let g be a bounding function. We say that A is *g-closed* if every string above which A is g-big is an element of A.

The concatenation property implies that every set A has a g-closure: the set of all strings above which A is g-big is g-closed.

Let $p \in \mathbb{P}_1$. We could require that B^p be b^p-closed by replacing it by its b^p-closure. In this case $T^p \backslash B^p$ is an $(h^p - b^p)$-bushy tree with no leaves.

In later sections we will use notions of largeness for which the concatenation property fails, and so will not be able to quite mimic this operation. Some amount of closure will be required to ensure that we get a restriction map from $\mathbb{P}_n$ to $\mathbb{P}_{n-1}$.

3.2.4 Totality

Recall that for a set of strings C we let $[C]^{\prec} = \bigcup_{\sigma \in C} [\sigma]^{\prec}$ be the set of $x \in \omega^{\omega}$ which extend some string in C.

Lemma 3.2.13. *Let $p \in \mathbb{P}_1$. Let $C \subseteq T^p$ be c.e. and open in T^p. Suppose that $p \Vdash x^G \in [C]^{\prec}$. Let $\tau \in T^p$; let $g \in Q$ such that $h^p \gg g$, and $h^p \geqslant g \geqslant b^p$ above $|\tau|$. Then the set $B^p \cup C$ is g-big above τ.*

Proof. Otherwise $q = (\tau, T^p \cap \tau^{\preccurlyeq}, (B^p \cup C) \cap \tau^{\preccurlyeq}, h^p, g)$ is a condition extending p which strongly forces that $x^G \notin [C]^{\prec}$. (We need $g \geqslant b^p$ above $|\tau|$ not to ensure that q is a condition but to ensure that it extends p.) □

Remark 3.2.14. Let $p \in \mathbb{P}_1$, let $C \subseteq T^p$ be c.e. and open in T^p, and suppose that p strongly forces that $x^G \in [C]^{\prec}$. By compactness there is some level m such that all strings in T^p of length m are in $B^p \cup C$. This shows that $B^p \cup C$ is h^p-big above every $\tau \in T^p$.

The following proposition shows that when we force totality of $\Gamma(x^G)$ (for some Turing functional Γ), we can in fact force strong totality.

Proposition 3.2.15. *Let $C \subseteq \omega^{\omega}$ be Π^0_2 and let $p \in \mathbb{P}_1$. Then $p \Vdash x^G \in C$ if and only if the set of conditions which strongly force that $x^G \in C$ is dense below p.*

Proof. It suffices to show that if $p \Vdash x^G \in C$ then p has an extension which strongly forces that $x^G \in C$. Fix such p.

By Lemma 3.1.6, find some $g \in Q$ such that $h^p \gg g \gg b^p$. As discussed above, every level of T^p contains a string above which $B^p \cap$

T^p is b^p-small. So by extending σ^p (and taking the full subtree above that string) we may assume that $h^p \geqslant g \geqslant b^p$ above $|\sigma^p|$.

Let $\langle C_k \rangle$ be a uniform sequence of c.e. subsets of T^p, open in T^p, such that $C \cap [T^p] = [T^p] \cap \bigcap_k [C_k]^{\prec}$. Lemma 3.2.13 says that for all $\tau \in T^p$, for all k, the set $B^p \cup C_k$ is g-big above τ.

We effectively define an increasing sequence $\langle S_k \rangle$ of finite g-bushy trees with the following properties:

- S_k is g-bushy;
- S_{k+1} is an end-extension of S_k, and no leaf of S_k is a leaf of S_{k+1};
- $S_k \subset T^p$; and
- the leaves of S_{k+1} lie in $B^p \cup C_k$.

We start with $S_0 = \{\sigma^p\}$. We know that $B^p \cup C_0$ is g-big above σ^p; Lemma 3.2.13 together with Lemma 3.2.7 shows that for all $k > 0$, $B^p \cup C_k$ is g-big above $B^p \cup C_{k-1}$. Thus, given S_k we can find a g-bushy end-extension S'_k of S_k with leaves in $B^p \cup C_k$; Remark 3.2.3 shows that $S'_k \subset T^p$. Since T^p has no leaves, we can extend S'_k to the required S_{k+1} by adding children from T^p to each leaf of S'_k (using the fact that $h^p \geqslant g$ above $|\sigma^p|$).

Having defined the trees S_k we let $S = \bigcup_k S_k$. Then $S \subseteq T^p$, S is g-bushy, and S has no leaves. Also, S is computable: a string of length k is in S if and only if it is in S_k.

Every path in S lies in $[B^p \cup C_k]^{\preccurlyeq}$ for all k and so $[S \backslash B^p] \subseteq C$. We required that $g \gg b^p$, so $q = (\sigma^p, S, B^p \cap S, g, b^p)$ is a condition which extends p and strongly forces that $x^G \in C$. □

3.2.5 Minimality

We prove:

Proposition 3.2.16. *Every condition in $\mathbb{P}_1$ forces that $\deg_{\mathrm{T}}(x^G)$ is minimal.*

Let $\Gamma\colon \omega^\omega \to 2^\omega$ be a Turing functional. There are three ways to ensure that $\Gamma(x^G)$ does not violate the minimality of $\deg_{\mathrm{T}}(x^G)$: ensuring that it is partial, ensuring that it is computable, or ensuring that it computes x^G.

For the rest of this section, fix a Turing functional $\Gamma\colon \omega^\omega \to 2^\omega$.

Definition 3.2.17. Let B be a set of strings. Two sets A_0 and A_1 of strings *Γ-split mod B* if $\Gamma(\tau_0) \perp \Gamma(\tau_1)$ for all $\tau_0 \in A_0 \backslash B$ and $\tau_1 \in A_1 \backslash B$.

Lemma 3.2.18. *Suppose that $p \in \mathbb{P}_1$ strongly forces that $\Gamma(x^G)$ is total, and forces that $\Gamma(x^G)$ is noncomputable.*

Let $\tau \in T^p$. Let $g \in \mathbf{Q}$ such that $h^p \gg g$, and $h^p \geqslant 3g$ and $g \geqslant b^p$ above $|\tau|$. Then there are $A_0, A_1 \subset T^p$, each g-big above τ, which Γ-split mod B^p.

Proof. Suppose that τ and g witness the failure of the lemma; we find an extension of p which forces that $\Gamma(x^G)$ is computable.

For $\alpha \in 2^{<\omega}$ let

$$A_\alpha = B^p \cup \{\rho \in T^p \ : \ \Gamma(\rho) \succcurlyeq \alpha\}$$

and

$$A_{\perp\alpha} = \bigcup A_\beta \ \ [\![\beta \in 2^{<\omega} \ \ \& \ \ \beta \perp \alpha]\!].$$

Let $\alpha \in 2^{<\omega}$ and suppose that A_α is $2g$-big above τ. By Remark 3.2.14 the set $A_{\alpha\hat{}0} \cup A_{\alpha\hat{}1}$ is h^p-big above every $\rho \in A_\alpha$. Since $h^p \geqslant 2g$ above $|\tau|$, the concatenation property implies that $A_{\alpha\hat{}0} \cup A_{\alpha\hat{}1}$ is $2g$-big above τ. By the big subset property there is some $i < 2$ such that $A_{\alpha\hat{}i}$ is g-big above τ [Here we use that the range of Γ is in Cantor rather than Baire space; we also use this in the proof of Lemma 3.2.20].

The assumption implies that $A_{\perp\alpha\hat{}i}$ is g-small above τ. Since $A_{\alpha\hat{}i} \cup A_{\perp\alpha\hat{}i}$ is h^p-big above τ and $3g \leqslant h^p$ above $|\tau|$ it must be that in fact $A_{\alpha\hat{}i}$ is $2g$-big above τ.

By recursion define the unique $z \in 2^\omega$ such that for all $\alpha \prec z$, A_α is $2g$-big above τ. Note that z is computable. The set

$$A_{\perp z} = \bigcup_{k<\omega} A_{\perp z\restriction_k}$$

is g-small above τ because it is the union of an increasing sequence of sets, each g-small above τ; since largeness is witnessed by a finite tree, g-smallness above τ is preserved when taking the union. The fact that z is computable shows that $A_{\perp z}$ is c.e., whence the tuple $(\tau, T^p \cap \tau^{\preccurlyeq}, A_{\perp z} \cap \tau^{\preccurlyeq}, h^p, g)$ is a condition extending p as required (recalling that $B^p \subseteq A_{\perp z}$). □

The following lemma will allow us to construct a "delayed splitting" subtree of T^p.

Lemma 3.2.19. *Suppose that $p \in \mathbb{P}_1$ strongly forces that $\Gamma(x^G)$ is total, and forces that $\Gamma(x^G)$ is noncomputable. Suppose that $\tau_1, \tau_2, \dots, \tau_k \in T^p$. Let $g \in \mathcal{Q}$ such that $h^p \gg g$, and $g \geqslant b^p$ and $h^p \geqslant 3^k g$ above $\min\{|\tau_1|, |\tau_2|, \dots, |\tau_k|\}$. Then there are sets $A_1, A_2, \dots, A_k \subset T^p$, each A_j g-big above τ_j, which pairwise Γ-split mod B^p.*

To prove Lemma 3.2.19 we need the following, which (mod B) is Lemma 6.2 of [32].

Lemma 3.2.20. *Let $g, h \in \mathcal{Q}$; let B be a set of strings. Suppose that:*

- *τ and τ^* are strings;*
- *A is a set of strings, $3g$-big above τ;*
- *For all $\rho \in A$, $E_{\rho,0}$ and $E_{\rho,1}$ are $3g$-big above ρ and Γ-split mod B; and*
- *F is a set of strings, $3h$-big above τ^*, satisfying $|\Gamma(\sigma)| > |\Gamma(\nu)|$ for all $\sigma \in F \backslash B$ and all $\nu \in E \backslash B$, where $E = \bigcup E_{\rho,i} \;\; [\![\rho \in A, i < 2]\!]$.*

Then there are $E' \subseteq E$, g-big above τ, and $F' \subseteq F$, h-big above τ^*, which Γ-split mod B.

We delay the proof of Lemma 3.2.20 until the end of the section.

Proof of Lemma 3.2.19, given Lemma 3.2.20. The proof is by induction on k. The lemma is vacuous for $k = 1$. Assume the lemma has been proven for k. Let $\tau_1, \ldots, \tau_k$ and τ^* be strings on T^p; suppose that $h^p \gg g$, and $h^p \geqslant 3^{k+1}g$ and $g \geqslant b^p$ above $\min\{|\tau^*|, |\tau_1|, |\tau_2|, \ldots, |\tau_k|\}$. The hypothesis for k holds for the bound $3g$ instead of g, and so by induction we find finite sets $A_1, \ldots, A_k \subset T^p$, each A_j $3g$-big above τ_j, which pairwise Γ-split mod B^p. As per Remark 3.2.8 we assume that $A_j \subset \tau_j^{\preccurlyeq}$.

For every $j = 1, \ldots, k$, for every $\rho \in A_j$, by Lemma 3.2.18 find finite $E_{\rho,0}$ and $E_{\rho,1}$, subsets of T^p, each $3g$-big above ρ and contained in $\rho^{\preccurlyeq}$, which Γ-split mod B^p. Let $E_j = \bigcup E_{\rho,i}\ [\![\rho \in A_j, i < 2]\!]$. Note that the E_j also pairwise Γ-split mod B^p.

Since $\bigcup_{j\leqslant k} E_j$ is finite, p strongly forces totality of $\Gamma(x^G)$ and $3^{k+1}g \leqslant h^p$ above $|\tau^*|$, by Remark 3.2.14 we find $F \subset T^p$ which is $3^k g$-big above τ^*, such that $|\Gamma(\sigma)| > |\Gamma(\nu)|$ for all $\sigma \in F \backslash B^p$ and $\nu \in \bigcup_{j\leqslant k} E_j \backslash B^p$.

Let $F_k = F$. By (reverse) recursion on $j = k, k-1, \ldots, 1$ we define sets $E'_j \subseteq E_j$ and $F_{j-1} \subseteq F_j$ such that every E'_j is g-big above τ_j, F_j is $3^j g$-big above τ^* and E'_j and F_{j-1} pairwise Γ-split mod B^p. To do this, given F_j apply Lemma 3.2.20 with $\tau = \tau_j$, $A = A_j$, g, τ^* and $E_{\rho,i}$ as themselves, $F = F_j$ and $h = 3^{j-1}g$.

In the end the sets E'_j for $j \leqslant k$ and F_0 are as required. □

Proposition 3.2.21. *Every condition in $\mathbb{P}_1$ forces that if $\Gamma(x^G)$ is total and noncomputable then $\Gamma(x^G) \equiv_T x^G$.*

Proof. It suffices to show that if $p \in \mathbb{P}_1$ forces that $\Gamma(x^G)$ is total and noncomputable then p has an extension which forces that $\Gamma(x^G) \equiv_T$

x^G. By Proposition 3.2.15 we may assume that p strongly forces that $\Gamma(x^G)$ is total.

By Lemma 3.1.6 find some $g \in \mathcal{Q}$ such that $h^p \gg g \gg b^p$. Let $\bar{g}(n) = \prod_{m<n} g(m)$. As above by extending σ^p we may assume that $h^p \geqslant 3^{\bar{g}} g$ and $g \geqslant b^p$ above $|\sigma^p|$ (see Example 3.1.4).

We effectively define an increasing sequence $\langle \ell_k \rangle$ and a sequence $\langle S_k \rangle$ of finite subtrees of T^p such that: (a) S_{k+1} is an end-extension of S_k; (b) the leaves of S_k all have length ℓ_k; and (c) S_k is *exactly* g-bushy: every nonterminal $\tau \in S_k$ has precisely $g(|\tau|)$ many immediate extensions on S_k.

Let $\ell_0 = |\sigma^p|$ and $S_0 = \{\sigma^p\}$. Suppose that S_k and ℓ_k have been defined. For every leaf τ of S_k we find a finite tree $R_\tau \subset T^p$, exactly g-bushy above τ, such that the sets of leaves of the various R_τ pairwise Γ-split mod B^p. This can be done since the number of leaves of S_k is $\prod_{m \in [|\sigma^p|, \ell_k)} g(m)$, which is bounded by $\bar{g}(\ell_k)$. We assumed that $h^p \geqslant 3^{\bar{g}} g$ and so $h^p \geqslant 3^{\bar{g}(\ell_k)} g$ above ℓ_k; so Lemma 3.2.19 applies.

Let S'_k be the union of S_k with the trees R_τ for all leaves τ of S_k. Let ℓ_{k+1} be greater than the height of S'_k; obtain S_{k+1} by appending a subtree of T^p, exactly g-bushy above ρ, to every leaf ρ of S'_k.

Let $S = \bigcup_k S_k$. As in the proof of Proposition 3.2.15, S is computable, computably bounded and has no leaves. It is g-bushy, and Γ is 1-1 on $[S \backslash B^p]$: if $x, x' \in [S \backslash B^p]$ and $x \upharpoonright_{\ell_k} \neq x' \upharpoonright_{\ell_k}$ then $\Gamma(x \upharpoonright_{\ell_{k+1}}) \perp \Gamma(x' \upharpoonright_{\ell_{k+1}})$. The tuple $(\sigma^p, S, B^p \cap S, g, b^p)$ is a condition as required (Lemma 3.1.8). □

Proof of Proposition 3.2.16. Let $p \in \mathbb{P}_1$. Let Γ be a Turing functional. If p has an extension which forces that $\Gamma(x^G)$ is partial then we are done. Otherwise p forces that $\Gamma(x^G)$ is total. We can extend p to a condition q which decides whether $\Gamma(x^G)$ is computable or not. If the former then we are done. Otherwise, Proposition 3.2.21 says that q forces that $\Gamma(x^G) \equiv_{\mathrm{T}} x^G$. □

Proof of Lemma 3.2.20. Let $E = \bigcup E_{\rho,i} \ \ [\![i < 2 \ \ \& \ \ \rho \in A]\!]$.

For a string $\alpha \in 2^{<\omega}$ let

$$F_{\succcurlyeq\alpha} = (F \cap B) \cup \{\sigma \in F \ : \ \Gamma(\sigma) \succcurlyeq \alpha\},$$

and similarly define $F_{\perp\alpha}$, $E_{\succcurlyeq\alpha}$, $E_{\preccurlyeq\alpha}$ and so on.

If $F \cap B$ is h-big above τ^* then we can let $F' = F \cap B$ and $E' = E$. Similarly if $E \cap B$ is g-big above τ.

Suppose otherwise. In that case, for sufficiently long α, $F_{\succcurlyeq\alpha}$ is h-small above τ^* (as it equals $F \cap B$). Let α be a string, maximal with respect to $F_{\succcurlyeq\alpha}$ being h-big above τ^*. We will show that either

1. $E_{\perp\alpha}$ is g-big above τ, or
2. $E_{\succcurlyeq\alpha}$ is g-big above τ and $F_{\perp\alpha}$ is h-big above τ^*.

In both cases we can find E' and F' as required.

We examine two cases, depending on $E_{\preccurlyeq\alpha}$.

First, suppose that $E_{\preccurlyeq\alpha}$ is g-big above τ. Let R be a tree witnessing this. Every leaf of R extends some element of A, so every element of R is comparable with some element of A. Since A is an antichain, the restriction of R to initial segments of elements of A is g-bushy. This shows that A', the set of $\rho \in A$ such that

$E_{\preccurlyeq\alpha}$ is g-big above ρ, is g-big above τ. We show that $E_{\perp\alpha}$ is g-big above every $\rho \in A'$; with the concatenation property this implies (1). Let $\rho \in A'$; there are two possibilities. If $B \cap E$ is g-big above ρ then we are done. Otherwise for some $i < 2$, $E_{\rho,i}$ intersects $E_{\preccurlyeq\alpha} \backslash B$. But then $E_{\rho,1-i} \subseteq E_{\perp\alpha}$, and $E_{\rho,1-i}$ is $3g$-big above ρ.

In the second case, suppose that $E_{\preccurlyeq\alpha}$ is g-small above τ. Since $E = E_{\perp\alpha} \cup E_{\succcurlyeq\alpha} \cup E_{\preccurlyeq\alpha}$ is $3g$-big above τ, either (1) holds, or $E_{\succcurlyeq\alpha}$ is g-big above τ. Assume the latter. We assumed that $E \cap B$ is g-small above τ; together, we see that $E_{\succcurlyeq\alpha} \backslash B$ is nonempty. In turn this implies that $|\Gamma(\sigma)| > |\alpha|$ for all $\sigma \in F \backslash B$; so $F = F_{\gneqq\alpha} \cup F_{\perp\alpha}$.

The maximality of α ensures that $F_{\gneqq\alpha}$ is $2h$-small above τ^* [Here again we use the fact that Γ maps into Cantor space]. Since F is $3h$-big above τ^* it must be that $F_{\perp\alpha}$ is h-big above τ^*, so (2) holds. □

3.3 A relatively DNC SMC of a DNC minimal degree

We now construct two sequences $x, y \in \omega^\omega$ such that $x \in \mathrm{DNC}$, x has minimal Turing degree, $y \in \mathrm{DNC}^x$ and $\deg_{\mathrm{T}}(x, y)$ is a strong minimal cover of $\deg_{\mathrm{T}}(x)$. Here, $\deg_{\mathrm{T}}(x, y)$ is the Turing degree of the computable join of x and y.

We use the mechanism of tree systems that was used by Cai [7, 6, 9] to show that there is a generalised high degree which is a minimal cover of a minimal degree. This is a more versatile approach than the homogenous trees which are usually used to construct initial segments of the Turing degrees (as in [35]).

3.3.1 Length 2 tree systems

Let $A \subseteq \omega^{<\omega} \times \omega^{<\omega}$ be a set of pairs of strings. For $\tau \in \omega^{<\omega}$ we let

$$A(\tau) = \{\rho \in \omega^{<\omega} \ : \ (\tau, \rho) \in A\}\,.$$

Of course $\operatorname{dom} A = \{\tau \ : \ \exists \rho \ [(\tau, \rho) \in A]\}$.

Definition 3.3.1. A *tree system* of length 2 above a pair (σ, μ) is a set T of pairs of strings satisfying:

- $\operatorname{dom} T$ is a tree above σ;
- For all $\tau \in \operatorname{dom} T$, $T(\tau)$ is a finite tree above μ; and
- If $\tau \prec \tau'$ are in $\operatorname{dom} T$ then $T(\tau')$ is an end-extension of $T(\tau)$.

In this section we only consider systems of length 2 and so we omit mentioning the length.

A tree system S is a subsystem of T if $S \subseteq T$. This means that $\operatorname{dom} S$ is a subtree of $\operatorname{dom} T$ and for all $\tau \in \operatorname{dom} S$, $S(\tau)$ is a subtree of $T(\tau)$. If $(\tau, \rho) \in T$ then $T \cap (\tau, \rho)^{\preccurlyeq}$ is a tree system, the system whose domain is the full subtree of $\operatorname{dom} T$ above τ and which maps all τ' in its domain to the full subtree of $T(\tau')$ above ρ. Here of course $(\tau, \rho)^{\preccurlyeq} = \tau^{\preccurlyeq} \times \rho^{\preccurlyeq}$ is the upwards-closure of $\{(\tau, \rho)\}$ in the partial ordering $\preccurlyeq$ on $(\omega^{<\omega})^2$ defined by the product of extension on strings: $(\tau, \rho) \preccurlyeq (\tau', \rho')$ if $\tau \preccurlyeq \tau'$ and $\rho \preccurlyeq \rho'$.

A tree system is *h-bounded* if for all $(\tau,\rho) \in T$, $\tau(n) < h(n)$ for all $n < |\tau|$ and $\rho(n) < h(n)$ for all $n < |\rho|$. It is *computably bounded* if it is bounded by some computable function.

If T is a computable and computably bounded tree system then $\operatorname{dom} T$ is computable and the map $\tau \mapsto T(\tau)$ is computable (for each $\tau \in \operatorname{dom} T$ we obtain a canonical index for $T(\tau)$ as a finite set).

Forest systems

To iterate largeness we require the notion of forest systems.

We call a set of pairs of strings $A \subset (\omega^{<\omega})^2$ *prefix-free* if $\operatorname{dom} A$ is prefix-free and for all $\tau \in \operatorname{dom} A$, $A(\tau)$ is prefix-free. For a set of pairs A let $A^{\preccurlyeq} = \bigcup_{(\sigma,\mu)\in A}(\sigma,\mu)^{\preccurlyeq}$ be the upwards closure of A under $\preccurlyeq$. If A is prefix-free then $A^{\preccurlyeq}$ is the *disjoint* union of $(\sigma,\mu)^{\preccurlyeq}$ for $(\sigma,\mu) \in A$. In other words, if (τ,ρ) extends some element of A then that element is unique. We denote this element by $(\tau,\rho)^{-A}$.

Definition 3.3.2. A *forest system* of length 2 above a finite prefix-free set $A \subset (\omega^{<\omega})^2$ is a set T of pairs of strings satisfying:

- $\operatorname{dom} T$ is a forest above $\operatorname{dom} A$;
- For all $\tau \in \operatorname{dom} T$, $T(\tau)$ is a finite forest above $A(\tau^{-\operatorname{dom} A})$ (where again $\tau^{-\operatorname{dom} A}$ is τ's unique predecessor in $\operatorname{dom} A$); and
- If $\tau \prec \tau'$ are in $\operatorname{dom} T$ then $T(\tau')$ is an end-extension of $T(\tau)$.

A *leaf* of a forest system T is a pair $(\tau,\rho) \in T$ such that τ is a leaf of $\operatorname{dom} T$ and ρ is a leaf of $T(\tau)$. Equivalently, it is a maximal element of the set of pairs T, if T is partially ordered by double extension $\preccurlyeq$. The set of leaves of a finite forest system is prefix-free.

Paths of tree systems

Let T be a tree system above (σ, μ). For $x \in [\operatorname{dom} T]$ we let

$$T(x) = \bigcup T(\tau) \ \ [\![\sigma \preccurlyeq \tau \prec x]\!].$$

We also let

$$[T] = \{(x, y) \ : \ x \in [\operatorname{dom} T] \ \ \& \ \ y \in [T(x)]\}.$$

In general the set $[T]$ need not be closed.

Lemma 3.3.3. *Suppose that for all $x \in [\operatorname{dom} T]$ the tree $T(x)$ has no leaves. Then $[T]$ is a closed subset of $[\sigma, \mu]^{\prec}$.*

Proof. For $\tau \in \operatorname{dom} T$ let

$$E_\tau = \bigcup [\rho]^{\prec} \ \ [\![\rho \text{ a leaf of } T(\tau)]\!];$$

for $n \geqslant |\sigma|$ let

$$E_n = \bigcup \left([\tau]^{\prec} \times E_\tau\right) \ \ [\![\tau \in \operatorname{dom} T \ \ \& \ \ |\tau| = n]\!].$$

Each E_n is clopen. We show that $[T] = \bigcap E_n$. We always have $[T] \subseteq \bigcap_{n \geqslant |\sigma|} E_n$. For suppose that $(x, y) \in [T]$, and let $n \geqslant |\sigma|$. Let $\tau = x\restriction_n$; so $\tau \in \operatorname{dom} T$. Let m be greater than the height of $T(\tau)$, and let $\rho = y\restriction_m$. Since $\rho \in T(x)$ there is some $\tau' \prec x$ such that $\rho \in T(\tau')$. Since $\rho \notin T(\tau)$ we must have $\tau \prec \tau'$, and so ρ extends some leaf of $T(\tau)$; this shows that $y \in E_\tau$, so $(x, y) \in E_n$.

In the other direction we use our assumption. Suppose that $(x, y) \in \bigcap_{n \geqslant |\sigma|} E_n$. For all $n \geqslant |\sigma|$, $(x, y) \in E_n$ implies that $x \upharpoonright_n \in \operatorname{dom} T$, so $x \in [\operatorname{dom} T]$. For all $n \geqslant |\sigma|$, some leaf of $T(x \upharpoonright_n)$ is an initial segment of y. To show that $y \in [T(x)]$ it suffices to show that the minimum length of a leaf in $T(x \upharpoonright_n)$ is unbounded as $n \to \infty$. But otherwise $T(x)$ would have a leaf. □

We will require that the pairs in tree systems appearing in our conditions can be extended to paths. It is not enough to require that the system does not have leaves.

Lemma 3.3.4. *Let T be a bounded tree system and suppose that* $\operatorname{dom} T$ *has no leaves. The following are equivalent:*

1. *For all k there is some m such that for every $\tau \in \operatorname{dom} T$ of length m, every leaf of $T(\tau)$ has length at least k.*
2. *For all $x \in [\operatorname{dom} T]$, $T(x)$ has no leaves.*

Proof. That (1) implies (2) is immediate. Suppose (2) holds. By Lemma 3.3.3, $[T]$ is closed; since T is bounded, $[T]$ is compact. Let $k < \omega$. The collection of clopen rectangles $[\tau, \rho]^{<}$ where $\tau \in \operatorname{dom} T$, ρ is a leaf of $T(\tau)$, and $|\rho| \geqslant k$ is an open cover of $[T]$; a finite sub-cover gives the desired m. □

To simplfy the combinatorics of finding big splittings, we restrict ourselves to "balanced" tree systems.

Definition 3.3.5. Let T be a tree system and let $n < \omega$. We say that m is a *balanced level of T* if for all $\tau \in \operatorname{dom} T$ of length m, every leaf of $T(\tau)$ has length m. We say that T is *balanced* if $\operatorname{dom} T$ has no leaves and T has infinitely many balanced levels.

If T is bounded and balanced then it satisfies the conditions of Lemma 3.3.4 and so by Lemma 3.3.3, $[T]$ is closed. If T is balanced, computable and computably bounded then $[T]$ is effectively closed (this is really where we use the requirement that if τ' extends τ in $\operatorname{dom} T$ then $T(\tau')$ is an end-extension, rather than any extension, of $T(\tau)$).

3.3.2 Bushiness for forest systems

Definition 3.3.6. Let g and h be bounding functions. A forest system T is *(g,h)-bushy* if $\operatorname{dom} T$ is g-bushy and for all $\tau \in \operatorname{dom} T$, $T(\tau)$ is h-bushy.

Lemma 3.3.7. *Let $A \subset (\omega^{<\omega})^2$ be finite and prefix-free, and let g and h be bounding functions. The following are equivalent for a set B of pairs of strings:*

1. *There is a finite (g,h)-bushy forest system above A, all of whose leaves lie in B.*
2. *The set of τ such that $B(\tau)$ is h-big above $A(\tau^{-\operatorname{dom} A})$ is g-big above* $\operatorname{dom} A$.

Proof. Assume (2). We define a forest system S by first defining $\operatorname{dom} S$, and then for all $\tau \in \operatorname{dom} S$, defining $S(\tau)$. We let $\operatorname{dom} S$ be a g-bushy forest above $\operatorname{dom} A$ such that for every leaf τ of $\operatorname{dom} S$,

$B(\tau)$ is h-big above $A(\tau^{-\operatorname{dom} A})$. Now let $\tau \in \operatorname{dom} S$; let $\sigma = \tau^{-\operatorname{dom} A}$. There are two cases. If τ is a leaf of $\operatorname{dom} S$ then we let $S(\tau)$ be an h-bushy forest above $A(\sigma)$ which witnesses that $B(\tau)$ is h-big above $A(\sigma)$. If τ is not a leaf of $\operatorname{dom} S$ then we let $S(\tau) = A(\sigma)$. □

These equivalent conditions define the notion of B being (g,h)-*big* above A; if they fail, we say that B is (g,h)-*small* above A. If A is infinite then we say that B is (g,h)-big above A if it is (g,h)-big above every finite prefix-free subset of A.

For brevity we let for $B \subseteq (\omega^{<\omega})^2$, a bounding function h and a finite prefix-free set of strings D

$$\pi^h_D(B) = \{\tau \,:\, B(\tau) \text{ is } h\text{-big above } D\}\,.$$

Note that $\pi^h_D(B) = \bigcap_{\rho \in D} \pi^h_\rho(B)$. A set B is (g,h)-big above a finite prefix-free set A if and only if for all $\sigma \in \operatorname{dom} A$, $\pi^h_{A(\sigma)}(B)$ is g-big above σ.

The big subset property (the analogue of Lemma 3.2.4) holds.

Lemma 3.3.8. *Let g, g' and h, h' be bounding functions and let $(\sigma,\mu) \in (\omega^{<\omega})^2$. Suppose that $B, C \subseteq (\omega^{<\omega})^2$ and that $B \cup C$ is $(g+g', h+h')$-big above (σ,μ). Then either B is (g,h)-big above (σ,μ) or C is (g',h')-big above (σ,μ).*

Proof. The set $\pi^{h+h'}_\mu(B \cup C)$ is $(g+g')$-big above σ. The big subset property implies that $\pi^{h+h'}_\mu(B \cup C) \subseteq \pi^h_\mu(B) \cup \pi^{h'}_\mu(C)$. Utilising the big subset property again, this time on the left coordinate, we see that either $\pi^h_\mu(B)$ is g-big above τ or $\pi^{h'}_\mu(C)$ is g'-big above τ. The first means that B is (g,h)-big above (σ,μ); the second, that C is (g',h')-big above (σ,μ). □

Weak concatenation

The concatenation property (Lemma 3.2.5) fails. Suppose that A is (g,h)-big above (σ,μ), and that B is (g,h)-big above every $(\tau,\rho) \in A$. It is possible that B is not (g,h)-big above (σ,μ): take for example two strings ρ_1 and ρ_2 and a string τ such that $(\tau,\rho_1),(\tau,\rho_2) \in A$. Then $\pi^h_{\rho_1}(B)$ and $\pi^h_{\rho_2}(B)$ are both g-big above τ, but the trees witnessing these facts need not be the same. That is, it is possible that $\pi^h_{\{\rho_1,\rho_2\}}(B)$ is g-small above τ. As a result, it is possible that a set B is (g,h)-small above some (σ,μ) but the set of pairs above which B is (g,h)-big is (g,h)-big above (σ,μ). Instead, we will employ a weak version of the concatenation property.

Definition 3.3.9. Let S and R be forest systems. We say that R is an *end-extension* of S if:

- $\operatorname{dom} R$ is an end-extension of $\operatorname{dom} S$;
- If $\tau \in \operatorname{dom} S$ is not a leaf of $\operatorname{dom} S$, then $R(\tau) = S(\tau)$;
- If τ is a leaf of $\operatorname{dom} S$ then $R(\tau)$ is an end-extension of $S(\tau)$.

Note that this relation is transitive. Now if T is a finite (length 1) forest above A, E is the set of leaves of T, and U is a forest above E, then $T \cup U$ is a forest above A, an end-extension of T whose leaves are the leaves of U. For forest systems we cannot take unions. Suppose that S is a finite forest system above A; let D be the set of leaves of S, and suppose that R is a forest system above D. We define the concatenation $S\hat{}R$ of S and R:

- $\operatorname{dom}(S\hat{}R) = (\operatorname{dom} S) \cup (\operatorname{dom} R)$;
- For $\tau \in \operatorname{dom} S \setminus \operatorname{dom} D$, $(S\hat{}R)(\tau) = S(\tau)$;
- For $\tau \in \operatorname{dom} R$, $(S\hat{}R)(\tau) = (S(\tau^{-\operatorname{dom} D})) \cup R(\tau)$.

This is a forest system above A, an end-extension of S whose leaves are the leaves of R. Note that if $\tau \in \operatorname{dom} D$ then we do not assume that $R(\tau) = D(\tau)$, and so it is possible that $(S\hat{}R)(\tau) \neq S(\tau)$. If both S and R are (g,h)-bushy then so is $S\hat{}R$. We conclude:

Lemma 3.3.10. *Suppose that B is (g,h)-big above A, and that C is (g,h)-big above B. Then C is (g,h)-big above A. Indeed, every finite (g,h)-bushy forest system whose leaves are in B has a finite (g,h)-bushy end-extension whose leaves are in C.*

A set B of pairs of strings is *open* if it is upwards closed in the partial ordering $\preccurlyeq$: closed under taking extensions in either coordinate.

The following lemma concerns sets of *strings*, not pairs of strings. It is a consequence of the concatenation property, and is formally proved by induction on $|\mathcal{B}|$.

Lemma 3.3.11. *Let $\mathcal{B}$ be a finite collection of open sets of strings, and let A be a finite, prefix-free set of strings. Suppose that each $B \in \mathcal{B}$ is g-big above every $\sigma \in A^{\preccurlyeq}$. Then $\bigcap \mathcal{B}$ is g-big above A.*

Lemma 3.3.12. *Let A and B be sets of pairs of strings, and let g and h be bounding functions. Suppose that B is open. Suppose that for all $(\sigma, \mu) \in A$, for all $\sigma' \succcurlyeq \sigma$, B is (g, h)-big above (σ', μ). Then B is (g, h)-big above A.*

Proof. It suffices to show that for any $\sigma \in \operatorname{dom} A$ and any finite, prefix-free $E \subseteq A(\sigma)$, $\pi^h_E(B)$ is g-big above σ. We apply Lemma 3.3.11 to the collection of sets $\pi^h_\mu(B)$ for $\mu \in E$. The fact that B is open implies that each $\pi^h_\mu(B)$ is open; the assumption is that each $\pi^h_\mu(B)$ is g-big above every extension of σ. □

Corollary 3.3.13 (Weak concatenation property)**.** *Let A, B and C be sets of pairs of strings, and suppose that C is open. Suppose that B is (g, h)-big above A, and that for all $(\tau, \rho) \in B$, for all $\tau' \succcurlyeq \tau$, C is (g, h)-big above (τ', ρ). Then C is (g, h)-big above A.*

Working within tree systems

We will need to apply the weak concatenation property while working within a given tree system T.

Remark 3.3.14. Suppose that B is (g, h)-big above A, that T is a tree system and that $A, B \subseteq T$. Then the forest system constructed in the proof of Lemma 3.3.7 is a subset of T.

Fix a tree system T. Suppose that S is a finite forest system; let D be the set of leaves of S. Let R be a forest system above D. Suppose that both S and R are subsets of T. Then $S\hat{\ }R$ is also a subset of T. Thus, Remark 3.3.14 can be extended. Suppose that B is (g, h)-big above A, that C is (g,h)-big above B, and that $A,B,C \subseteq T$. Then not only is there a finite (g,h)-bushy forest system $S \subseteq T$ above A whose leaves are in B, but further, any such system S can be end-extended to a finite (g,h)-bushy forest system $R \subseteq T$ above A whose leaves are in C.

If T is a tree system and $B \subseteq T$ then we say that B is *open in T* if it is upwards closed in the restriction of the partial ordering $\preccurlyeq$ to T. Lemma 3.3.11 can be "restricted to a tree S": if $A,B \subseteq S$ and each $B \in \mathcal{B}$ is open in S and g-big above $A^{\preccurlyeq} \cap S$, then $\bigcap \mathcal{B}$ is g-big above A. We then obtain a version of Lemma 3.3.12 restricted to T:

Lemma 3.3.15. *Let T be a tree system; let $A,B \subseteq T$, and let g and h be bounding functions. Suppose that B is open in T, and that for all $(\sigma,\mu) \in A$, for all $\sigma' \succcurlyeq \sigma$ in $\operatorname{dom} T$, B is (g,h)-big above (σ',ρ). Then B is (g,h)-big above A.*

And so we get the weak concatenation property within a tree system:

Corollary 3.3.16. *Let T be a tree system, let $A,B,C \subseteq T$, and suppose that C is open in T. Suppose that B is (g,h)-big above A, and that for all $(\tau,\rho) \in B$, for all $\tau' \succcurlyeq \tau$ in $\operatorname{dom} T$, C is (g,h)-big above (τ',ρ). Then C is (g,h)-big above A, and in fact every finite (g,h)-bushy forest system $S \subseteq T$ which witnesses that B is (g,h)-big above A has an end-extension $R \subseteq T$ which witnesses that C is (g,h)-big above A.*

We obtain a lemma which will allow us to take full subsystems as extensions.

Lemma 3.3.17. *Let T be a bounded and balanced (b,c)-bushy tree system above (σ,μ) and let $B \subset T$ be open in T and (b,c)-small above (σ,μ). Then for every m there is some $(\tau,\rho) \in T$ such that $|\tau|, |\rho| \geqslant m$ and above which B is (b,c)-small.*

Proof. Let m be some balanced level of T. Let D be the set of pairs $(\tau,\rho) \in T$ such that $|\tau| = |\rho| \geqslant m$. Then D is (b,c)-big above (σ,μ). If there is no pair as required then the weak concatenation property localised to T (Corollary 3.3.16) shows that B is (b,c)-big above (σ,μ). □

Remark 3.3.18. We use the same convention discussed in Remark 3.2.8; we assume that large sets given to us are sets of leaves of tree systems witnessing their largeness. For example, if we are given a set B of pairs, (g,h)-big above some A, then we assume that B is finite and prefix-free; that for all $\tau \in \operatorname{dom} B$, $B(\tau)$ is h-big above $A(\tau^{-\operatorname{dom} A})$; and that $B \subseteq A^{\preccurlyeq}$.

3.3.3 The notion of forcing and the generic

Let B_{DNC^2} be the set of pairs (τ,ρ) such that $\tau \in B_{\mathrm{DNC}}$ or $\rho \in B_{\mathrm{DNC}^\tau}$; the latter means that $\rho(e) = J^\tau(e)\downarrow$ for some $e < |\rho|$. Note that this set of pairs is $(2,2)$-small above $(\langle\rangle,\langle\rangle)$.

We let $\mathbb{P}_2$ be the set of tuples $p = ((\sigma^p,\mu^p), T^p, B^p, h^p, b^p)$ satisfying:

1. T^p is a computably bounded, computable, balanced tree system above (σ^p, μ^p);
2. $h^p \in Q$ and T^p is (h^p, h^p)-bushy;
3. $B^p \subset T^p$ is c.e. and open in T^p, and $B^p \supseteq B_{\mathrm{DNC}^2} \cap T^p$;
4. $b^p \in Q$ and B^p is (b^p, b^p)-small above (σ^p, μ^p); and
5. $h^p \gg b^p$ and $h^p \geqslant b^p$ above $\min\{|\sigma^p|, |\mu^p|\}$.

We define a partial ordering on $\mathbb{P}_2$ as follows. A condition q extends a condition p if $(\sigma^p, \mu^p) \preccurlyeq (\sigma^q, \mu^q)$, T^q is a subsystem of T^p, $B^p \cap T^q \subseteq B^q$, and $h^q \leqslant h^p$ and $b^q \geqslant b^p$ above $\min\{|\sigma^q|, |\mu^q|\}$.

Lemma 3.3.19. *The assignment of closed sets $X^p = [T^p] \backslash [B^p]^{\prec}$ for $p \in \mathbb{P}_2$ is acceptable (Definition 3.1.10).*

Note that $T^p \backslash B^p$ may not be a tree system and so we have not defined $[T^p \backslash B^p]$.

Proof. As discussed above, the fact that T^p is balanced implies that $[T^p]$ is closed. That $X^q \subseteq X^p$ when q extends p again follows directly from the definition of the partial ordering on $\mathbb{P}_2$.

Let $p \in \mathbb{P}_2$. Suppose that $[T^p] \subseteq [B^p]^{\prec}$. Since T^p is bounded, $[T^p]$ is compact. There is some finite $C \subset B^p$ such that $[T^p] \subseteq [C]^{\prec}$. We may assume that C is prefix-free. Then C shows that B^p is (h^p, h^p)- and so (b^p, b^p)-big above (σ^p, μ^p). Hence X^p is nonempty.

Let $m < \omega$. Since $h^p \geqslant b^p$ above $\min\{|\sigma^p|, |\mu^p|\}$ Lemma 3.3.17 shows that there is some pair $(\tau, \rho) \in T^p$ with $|\tau|, |\rho| \geqslant m$ above which B^p is (b^p, b^p)-small. Then $q = ((\tau,\rho), T^p \cap (\tau,\rho)^{\preccurlyeq}, B^p \cap (\tau,\rho)^{\preccurlyeq}, h^p, b^p)$ is a condition in $\mathbb{P}_2$ extending p satisfying $X^q \subseteq [T^q] \subseteq [\tau,\rho]^{\prec}$. Thus for every m, the set of conditions $p \in \mathbb{P}_2$ such that $X^p \subseteq [\tau,\rho]^{\prec}$ for some strings τ,ρ, both of length at least m, is dense in $\mathbb{P}_2$; this implies requirement (c) of Definition 3.1.10. □

As in the previous section, if $G \subset \mathbb{P}_2$ is sufficiently generic then $\bigcap_{p\in G}[T^p]\backslash[B^p]^{\prec}$ is a singleton which we denote by $\{(x^G, y^G)\}$. In fact $x^G = \bigcup\{\sigma^p : p \in G\}$ and $y^G = \bigcup\{\mu^p : p \in G\}$.

Let $p \in \mathbb{P}_2$; since $B_{\mathrm{DNC}^2} \subseteq B^p$ we see:

Proposition 3.3.20. *Every condition in $\mathbb{P}_2$ forces that $x^G \in$ DNC and that $y^G \in \mathrm{DNC}^{x^G}$.*

The restriction of $\mathbb{P}_2$ to $\mathbb{P}_1$

We do not actually have a restriction map to $\mathbb{P}_1$ from $\mathbb{P}_2$ but from a dense subset of $\mathbb{P}_2$. Note that if $\mathbb{Q} \subseteq \mathbb{P}$ is dense and $G \subset \mathbb{Q}$ is a generic directed set, then it is also a generic directed subset of $\mathbb{P}$.

Proposition 3.3.21. *There is a dense subset $\mathbb{Q}_2 \subseteq \mathbb{P}_2$ and a restriction map $i\colon \mathbb{Q}_2 \to \mathbb{P}_1$ such that for all $p \in \mathbb{Q}_2$, $X^{i(p)} \supseteq \operatorname{dom} X^p$.*

In particular this shows that $\mathbb{P}_2$ is nonempty.

Proof. We define $i\colon \mathbb{P}_2 \to \mathbb{P}_1$ by letting

$$i(q) = (\sigma^q, \operatorname{dom} T^q, \pi^{b^q}_{\mu^q}(B^q), h^q, b^q)$$

where we recall that $\pi^{b^q}_{\mu^q}(B^q)$ is the set of τ such that $B^q(\tau)$ is b^q-big above μ^q.

Let $q \in \mathbb{P}_2$. It is routine to check that $i(q) \in \mathbb{P}_1$.

However, i is not order-preserving. For this reason we let

$$\mathbb{Q}_2 = \left\{ p \in \mathbb{P}_2 \, : \, \pi^{b^p}_{\mu^p}(B^p) = \{\tau \in \operatorname{dom} T^p \, : \, \mu^p \in B^p(\tau)\} \right\}.$$

Suppose that $q \in \mathbb{Q}_2$; then $X^{i(q)} \supseteq \operatorname{dom} X^q$. To check this we observe that if $(x,y) \in [T^q]\backslash[B^q]^{<}$ then for all $\tau < x$, $(\tau, \mu^q) \notin B^q$ and so $\tau \notin B^{i(q)}$; so $x \in [\operatorname{dom} T^q]\backslash[B^{i(q)}]^{<}$. (In fact $X^{i(q)} = \operatorname{dom} X^q$; if $x \in X^{i(q)}$ then $B^q(x)$ is b^q-small above μ^q, so $T^q(x)\backslash B^q(x)$ has a path.)

Let $q \in \mathbb{P}_2$. Define a set $B \subset T^q$: for $\tau \in \operatorname{dom} T^q \backslash \pi^{b^q}_{\mu^q}(B^q)$ we let $B(\tau) = B^q(\tau)$; for $\tau \in \pi^{b^q}_{\mu^q}(B^q)$ we let $B(\tau) = T^q(\tau)$. Let $\nu(q) = ((\sigma^q, \mu^q), T^q, B, h^q, b^q)$. The concatenation property implies that $\pi^{b^q}_{\mu^q}(B^q) = \pi^{b^q}_{\mu^q}(B)$, which shows that $\nu(q) \in \mathbb{P}_2$, in fact that $\nu(q) \in \mathbb{Q}_2$, and it extends q. Hence $\mathbb{Q}_2$ is dense in $\mathbb{P}_2$. We observe that $i(q) = i(\nu(q))$.

To show that the restriction of i to $\mathbb{Q}_2$ is order-preserving we need to check that if $q, s \in \mathbb{Q}_2$ and q extends s, then $B^{i(s)} \cap T^{i(q)} \subseteq B^{i(q)}$. If $\tau \in B^{i(s)}$ (and $\tau \in T^{i(q)}$) then $(\tau, \mu^s) \in B^s$; since B^s is open in T^s, this means that $(\tau, \mu^q) \in B^s$; since $B^s \cap T^q \subseteq B^q$, $(\tau, \mu^q) \in B^q$ and so $\tau \in B^{i(q)}$.

Let $q \in \mathbb{Q}_2$ and let $p \in \mathbb{P}_1$ extend $i(q)$; we need to find $r \in \mathbb{Q}_2$ extending q such that $i(r)$ extends p. Using the map ν, it suffices to find $r \in \mathbb{P}_2$.

Let T be the restriction of T^q to T^p: $\operatorname{dom} T = T^p$ and for $\tau \in T^p$, $T(\tau) = T^q(\tau)$. The system T is (h^p, h^q)-bushy above (σ^p, μ^q).

Also define $B \subseteq T$; if $\tau \in B^p$ then $B(\tau) = T(\tau)$; if $\tau \in T^p \backslash B^p$ then

$B(\tau) = B^q(\tau)$. The set B is open in T, is c.e., and is (b^p, b^q)-small above (σ^p, μ^q). To see that let S be (b^p, b^q) bushy above (σ^p, μ^q); by Remark 3.3.14 we may assume that $S \subset T$. Since $\operatorname{dom} S$ is a subtree of T^p we find a leaf τ of $\operatorname{dom} S$ which is not in B^p. Since p extends $i(q)$, $\tau \notin B^{i(q)}$ and so $B(\tau) = B^q(\tau)$ is b^q-small above μ^q, so $S(\tau)$ has a leaf ρ which is not in $B(\tau)$.

Since $h^p \geqslant b^p$ above $|\sigma^p|$ and $h^q \geqslant b^q$ above $|\mu^q|$, T is (b^p, b^q)-bushy. By Lemma 3.3.17 we can find $(\sigma, \mu) \in T$ such that $|\sigma|, |\mu| \geqslant \max\{|\sigma^p|, |\mu^q|\}$ and above which B is (b^p, b^q)-small.

We now define $r = ((\sigma, \mu), T \cap (\sigma, \mu)^{\preccurlyeq}, B \cap (\sigma, \mu)^{\preccurlyeq}, h^p, b^p)$. The point is that $h^p \leqslant h^q$ and $b^p \geqslant b^q$ above $\min\{|\sigma|, |\mu|\}$ and so T^r is (h^p, h^p)-bushy and B^r is (b^p, b^p)-small above (σ, μ). This also shows that r extends q. To show that $i(r)$ extends p we need to show that $B^p \cap \operatorname{dom} T^r \subseteq B^{i(r)}$. Let $\tau \in B^p \cap \operatorname{dom} T^r$. Then $\tau \succcurlyeq \sigma$ and so $\mu \in T(\tau) = B(\tau)$, so $\tau \in B^{i(r)}$. $\square$

Corollary 3.3.22. *Every condition in $\mathbb{P}_2$ forces that x^G has minimal Turing degree.*

Totality

Proposition 3.3.23. *Let $C \subseteq (\omega^\omega)^2$ be Π^0_2 and let $p \in \mathbb{P}_2$. If $p \Vdash (x^G, y^G) \in C$ then p has an extension which strongly forces that $(x^G, y^G) \in C$.*

Proof. The proof is similar to the proof of Proposition 3.2.15. We choose a function $g \in Q$ such that $h^p \gg g \gg b^p$. By Lemma 3.3.17 we may assume that $h^p \geqslant g \geqslant b^p$ above $\min\{|\sigma^p|, |\mu^p|\}$.

We fix a sequence of c.e. sets $C_k \subseteq T^p$, open in T^p, such that $C \cap [T^p] = [T^p] \cap \bigcap_k [C_k]^{\prec}$. For all $(\tau, \rho) \in T^p$, for all k, the set $B^p \cup C_k$ is (g, g)-big above (τ, ρ); otherwise $((\tau, \rho), T^p \cap (\tau, \rho)^{\preccurlyeq}, (B^p \cup C_k) \cap (\tau, \rho)^{\preccurlyeq}, h^p, g)$ is a condition extending p which forces that $(x^G, y^G) \notin C$.

We define a sequence of finite tree systems $S_k \subset T^p$ such that: each S_k is (g, g)-bushy; S_{k+1} is a proper end-extension of S_k; the leaves of S_{k+1} are in $C_k \cup B^p$; if $k > 0$ then there is some ℓ_k such that for every $k \geqslant 1$, for every leaf (τ, ρ) of S_k, $|\tau| = |\rho| = \ell_k$. We begin with $S_0 = \{(\sigma^p, \mu^p)\}$. Given S_k, Corollary 3.3.16 says that $C_k \cup B^p$ is (g, g)-big above the set of leaves of S_k, so we can find a finite (g, g)-bushy end-extension $S'_k \subset T^p$ of S_k with leaves in $C_k \cup B^p$.

Now find some ℓ_{k+1}, greater than $|\tau|$ and $|\rho|$ for any leaf (τ, ρ) of S'_k, which is a balanced level for T^p (Definition 3.3.5). Then the set of $(\tau, \rho) \in T^p$ such that $|\tau| = |\rho| = \ell_{k+1}$ is (g, g)-big above the set of leaves of S'_k. Hence we can find $S_{k+1} \subset T^p$ to be an end-extension of S'_k as required.

It follows that $S = \bigcup_k S_k$ is a computable, (g, g)-bushy and balanced tree system above (σ^p, μ^p) and that the condition $((\sigma^p, \mu^p), S, B^p \cap S, g, b^p)$ extends p and strongly forces that $(x^G, y^G) \in C$. $\square$

3.3.4 Minimal cover

We work toward showing that $\deg_{\mathrm{T}}(x^G, y^G)$ is a strong minimal cover of $\deg_{\mathrm{T}}(x^G)$. We do this in two steps. First we show that it is a minimal cover. This mostly uses the tools of the previous section.

Let $\Gamma\colon (\omega^\omega)^2 \to 2^\omega$ be a Turing functional. For a condition $p \in \mathbb{P}_2$, a bounding function g and a string μ let $\Gamma\text{-Sp}^g_\mu(p)$ be the set of $\tau \in \operatorname{dom} T^p$ such that $T^p(\tau)$ contains two sets $A_0(\tau)$ and $A_1(\tau)$, both g-big above μ, which $\Gamma(\tau,-)$-split mod $B^p(\tau)$.

Lemma 3.3.24. *Suppose that $p \in \mathbb{P}_2$ strongly forces that $\Gamma(x^G, y^G)$ is total and forces that $\Gamma(x^G, y^G) \not\leqslant_{\mathrm{T}} x^G$. Let $(\sigma,\mu) \in T^p$. Let $g \in Q$ such that $h^p \gg g$, and $h^p \geqslant 3g$ and $g \geqslant b^p$ above* $\min\{|\sigma|, |\mu|\}$. *Then $\Gamma\text{-Sp}^g_\mu(p)$ is g-big above σ.*

Proof. Suppose that (σ,μ) and g witness the failure of the lemma; we find an extension of p which forces that $\Gamma(x^G, y^G)$ is computable from x^G.

Let Θ be the (c.e.) set of pairs (τ,α) such that $\tau \in \operatorname{dom} T^p$, $\alpha \in 2^{<\omega}$ and $A_\alpha(\tau)$ is g-big above μ, where as before $A_\alpha = B^p \cup \{(\tau,\rho) \in T^p : \Gamma(\tau,\rho) \geqslant \alpha\}$.

For brevity let $C = \Gamma\text{-Sp}^g_\mu(p)$. The set C is open in $\operatorname{dom} T^p$. If $\tau \in \operatorname{dom} T^p \backslash C$ then the strings in $\Theta(\tau)$ are pairwise comparable.

Let $\tau \in \operatorname{dom} T^p \backslash C$. The argument of the proof of Lemma 3.2.18 shows that if $|\Gamma(\tau,\rho)| \geqslant m$ for every leaf ρ of $T^p(\tau)$ which is not in $B^p(\tau)$ then $\Theta(\tau)$ contains a string of length m. Also, $B^p(\tau)$ is g-small above μ and so $\Theta(\tau)$ is finite; in this case we let $\Theta^\tau = \bigcup \Theta(\tau)$ be the longest string in $\Theta(\tau)$.

If $\tau \preccurlyeq \tau'$ are in $\operatorname{dom} T^p \backslash C$ then $\Theta^\tau \preccurlyeq \Theta^{\tau'}$. This follows from the fact that $A_\alpha(\tau) \subseteq A_\alpha(\tau')$ for all α.

Let $D = \{(\tau,\rho) \in T^p : \tau \in C$ or $\Gamma(\tau,\rho) \perp \alpha$ for some $\alpha \in$

$\Theta(\tau)\}$. The set D is c.e. and is open in T^p. Also, $D \cup B^p$ is (g,g)-small above (σ,μ). To see this, suppose that $S \subset T^p$ is a finite (g,g)-bushy tree system above (σ,μ) (as above we use Remark 3.3.14). Then there is a leaf τ of $\operatorname{dom} S$ which is not in C; and then $S(\tau)$ must contain a leaf $\rho \notin B^p(\tau)$ such that $\Gamma(\tau,\rho)$ is compatible with Θ^τ.

Now suppose that $(x,y) \in [T^p]\backslash[D \cup B^p]^{\preccurlyeq}$. No initial segment of x is in C. A compactness argument shows that $\Theta(x) = \bigcup_{\tau<x} \Theta^\tau$ is total, and so $\Gamma(x,y) = \Theta(x)$. Certainly $\Theta(x) \leqslant_{\mathrm{T}} x$. Therefore the condition $((\sigma,\mu), T^p \cap (\sigma,\mu)^{\preccurlyeq}, (D \cup B^p) \cap (\sigma,\mu)^{\preccurlyeq}, h^p, g)$ extends p and (strongly) forces that $\Gamma(x^G, y^G) \leqslant_{\mathrm{T}} x^G$. □

Definition 3.3.25. Let $B \subseteq (\omega^{<\omega})^2$. Two sets A_0 and A_1 of pairs of strings *locally* Γ-*split mod* B if for all τ, $A_0(\tau)$ and $A_1(\tau)$ form a $\Gamma(\tau,-)$-splitting mod $B(\tau)$. That is, if $(\tau,\rho_0) \in A_0\backslash B$ and $(\tau,\rho_1) \in A_1\backslash B$ then $\Gamma(\tau,\rho_0) \perp \Gamma(\tau,\rho_1)$.

We introduce the notion of uniform largeness.

Definition 3.3.26. Let A be finite and prefix-free, and let $\mathcal{B}$ be a collection of sets of pairs of strings. We say that the sets in $\mathcal{B}$ are *uniformly* (g,h)-*big above* A if the set of τ such that for all $B \in \mathcal{B}$, $B(\tau)$ is h-big above $A(\tau^{-\operatorname{dom} A})$, is g-big above $\operatorname{dom} A$.

The conclusion of Lemma 3.3.24 is that there are A_0 and A_1, subsets of T^p uniformly (g,g)-big above (σ,μ), which locally Γ-split mod B^p.

Lemma 3.3.27. *Suppose that $p \in \mathbb{P}_2$ strongly forces that $\Gamma(x^G, y^G)$ is total and forces that $\Gamma(x^G, y^G) \not\leqslant_T x^G$.*

Let $\sigma \in \operatorname{dom} T^p$, and let $\mu_1, \mu_2, \ldots, \mu_k$ be elements of $T^p(\sigma)$. Let $g \in Q$ such that $h^p \gg g$, and $h^p \geqslant 3^k g$ and $g \geqslant b^p$ above $\min\{|\sigma|, |\mu_1|, |\mu_2|, \ldots, |\mu_k|\}$. Then there is a set $A \subset T^p$, (g,g)-big above $\{(\sigma, \mu_j) : j \leqslant k\}$, such that the sets $A \cap (\sigma, \mu_j)^{\preccurlyeq}$ pairwise locally Γ-split mod B^p.

Proof. The idea is to extend bushily on the first coordinate so that we can emulate the proof of Lemma 3.2.19 on the second coordinate. Formally this is done by induction on k. Suppose this has been shown for k; let $\mu_1, \ldots, \mu_k$ and μ^* be elements of $T^p(\sigma)$; suppose that $h \gg g$, and $h^p \geqslant 3^{k+1} g$ and $g \geqslant b^p$ above $\min\{|\sigma|, |\mu^*|, |\mu_j| : j \leqslant k\}$. Then $h \gg 3g$; so by induction we can find a set A, $(3g, 3g)$-big above $\{(\sigma, \mu_j) : j \leqslant k\}$ such that the sets $A \cap (\sigma, \mu_j)^{\preccurlyeq}$ pairwise locally Γ-split mod B^p. In fact we only need $(g, 3g)$-big.

Let $(\zeta, \nu) \in A$. By Lemma 3.3.24, for all $\zeta' \succcurlyeq \zeta$ on $\operatorname{dom} T^p$, $\Gamma\text{-}\mathrm{Sp}^{3g}_\nu(p)$ is $3g$-big above ζ' (again we only need g-big). By repeatedly extending we see that for all $\zeta \in \operatorname{dom} A$, $Q_\zeta = \zeta^{\preccurlyeq} \cap \bigcap_{\nu \in A(\zeta)} \Gamma\text{-}\mathrm{Sp}^{3g}_\nu(p)$ is $3g$-big above ζ. We extend the set A by letting $A(\tau) = A(\zeta)$ for all $\tau \in Q_\zeta$. Let $Q = \bigcup_{\zeta \in \operatorname{dom} A} Q_\zeta$; it is $3g$-big above σ. For every $\tau \in Q$ and all $\nu \in A(\tau)$ we can find sets $E_{\nu,0}(\tau), E_{\nu,1}(\tau) \subset T^p(\tau)$, each $3g$-big above ν, which $\Gamma(\tau, -)$-split mod $B^p(\tau)$.

Further, by extending in $\operatorname{dom} T^p$, we may assume that for all $\tau \in Q$ we can find $F(\tau) \subset T^p(\tau)$ which is $3^k g$-big above μ^* and such that $|\Gamma(\tau, \rho)| > |\Gamma(\tau, \eta)|$ for all $\rho \in F(\tau) \backslash B^p(\tau)$ and all $\eta \in E_{\nu,i}(\tau)$

(for both $i < 2$ and all $\nu \in A(\tau)$).

Overall we see that for all $\tau \in Q$ we can run the argument proving Lemma 3.2.19 inside $T^p(\tau)$ and using Lemma 3.2.20 find $F'(\tau) \subseteq F(\tau)$, g-big above μ^* and for $j \leqslant k$, $E'_j(\tau) \subset T^p(\tau)$, g-big above μ_j, with every string in $E'_j(\tau)$ extending some string in $A_j(\tau)$, such that $F'(\tau)$ and $E'_j(\tau)$ form a $\Gamma(\tau, -)$-splitting mod $B^p(\tau)$; the fact that strings in $E'_j(\tau)$ extend strings in $A(\tau) \cap \mu_j^{\leqslant}$ shows that the sets $E'_j(\tau)$ also pairwise $\Gamma(\tau, -)$-split mod $B^p(\tau)$. □

Proposition 3.3.28. *Every condition in $\mathbb{P}_2$ forces that if $\Gamma(x^G, y^G)$ is total and $\Gamma(x^G, y^G) \nleqslant_T x^G$ then $\Gamma(x^G, y^G) \oplus x^G \geqslant_T y^G$.*

Proof. As in the proof of Proposition 3.2.21 we take some $p \in \mathbb{P}_2$ which strongly forces that $\Gamma(x^G, y^G)$ is total and forces that $\Gamma(x^G, y^G) \nleqslant_T x^G$, and find an extension of p which forces that $\Gamma(x^G, y^G) \oplus x^G \geqslant_T y^G$.

Find some $g \in Q$ such that $h^p \gg g \gg b^p$. Let $\bar{g}(m) = \prod_{k<m} g(k)$. By Lemma 3.3.17 we can extend (σ^p, μ^p) so that $h^p \geqslant 3^{\bar{g}} g$ and $g \geqslant b^p$ above $\min\{|\sigma^p|, |\mu^p|\}$.

We define an increasing sequence $\langle \ell_k \rangle$ and a sequence $\langle S_k \rangle$ of finite subsystems of T^p such that: $\operatorname{dom} S_k$ is g-bushy and for all $\tau \in \operatorname{dom} S_k$, $S_k(\tau)$ is exactly g-bushy; S_{k+1} is a proper end-extension of S_k; for every leaf (τ, ρ) of S_k, $|\tau| = |\rho| = \ell_k$.

To begin we find some $\ell_0 > |\sigma^p|, |\mu^p|$, a balanced level for T^p. We let $\operatorname{dom} S_0 = \operatorname{dom} T^p \upharpoonright_{\omega^{\leqslant \ell_0}}$ and for each leaf τ of $\operatorname{dom} S_0$ we let $S_0(\tau)$ be an exactly g-bushy subtree of $T^p(\tau)$ whose leaves all have lenght ℓ_0. As usual if $\tau \in \operatorname{dom} S_0$ is not a leaf then we let

$S_0(\tau) = \{\mu^p\}$.

Given S_k we note that for every leaf σ of $\operatorname{dom} S_k$, the number of leaves of $S_k(\sigma)$ is precisely $\prod_{m\in[|\mu^p|,\ell_k)} g(m)$ which is bounded by $\bar{g}(\ell_k)$; and $h^p \geqslant 3^{\bar{g}(\ell_k)}g$ above ℓ_k. By Lemma 3.3.27 we can find for each leaf σ of $\operatorname{dom} S_k$ a finite (g,g)-bushy forest system $R_\sigma \subset T^p$ above $\{(\sigma,\nu) : \nu \text{ a leaf of } S_k(\sigma)\}$, such that for every leaf τ of $\operatorname{dom} R_\sigma$, the sets $R_\sigma(\tau) \cap \nu^{\preccurlyeq}$ for the leaves ν of $S_k(\sigma)$ pairwise $\Gamma(\tau,-)$-split mod B^p. By shrinking we may assume that for all leaves $\tau \in \operatorname{dom} R_\sigma$, $R_\sigma(\tau)$ is exactly g-bushy. Let $R = \bigcup_\sigma R_\sigma$ and let $S'_k = S_k\hat{\ }R$.

Now as in the proof of Proposition 3.3.23 we let ℓ_{k+1} be a balanced level of T^p, greater than the length of any string appearing in S'_k, and let $S_{k+1} \subset T^p$ be an end-extension of S'_k with the desired properties.

Let $S = \bigcup_k S_k$. Then for all $x \in [\operatorname{dom} S]$, $\Gamma(x,-)$ is 1-1 on $[S(x)]\backslash[B^p(x)]^{\prec}$. The tuple $((\sigma^p,\mu^p), S, B^p \cap S, g, b^p)$ is a condition as required (relativise Lemma 3.1.8 to each x). $\square$

3.3.5 Strong minimal cover

The following is the usual definition of splitting, restated for pairs of strings.

Definition 3.3.29. Let $B \subseteq (\omega^{<\omega})^2$. Two sets A_0 and A_1 Γ-*split mod* B if for all $(\tau,\rho) \in A_0\backslash B$ and $(\tau',\rho') \in A_1\backslash B$, $\Gamma(\tau,\rho) \perp \Gamma(\tau',\rho')$.

Lemma 3.3.30. *Let $g_1, g_2, h_1, h_2 \in Q$; let B be an open set of pairs of strings. Suppose that:*

- *(σ, μ) and (σ^*, μ^*) are pairs of strings;*
- *A is $(3g_1, 3g_2)$-big above (σ, μ);*
- *E_0 and E_1 are uniformly $(3g_1, 3g_2)$-big above A; and for all $(\tau, \rho) \in A$, $E_0 \cap (\tau, \rho)^{\preccurlyeq}$ and $E_1 \cap (\tau, \rho)^{\preccurlyeq}$ locally Γ-split mod B; and*
- *F is $(3h_1, 3h_2)$-big above (σ^*, μ^*), and $|\Gamma(\lambda, \nu)| > |\Gamma(\zeta, \eta)|$ for all $(\lambda, \nu) \in F \backslash B$ and all $(\zeta, \eta) \in E \backslash B$, where $E = E_0 \cup E_1$.*

Then there are $E' \subseteq E$, (g_1, g_2)-big above (σ, μ), and $F' \subseteq F$, (h_1, h_2)-big above (σ^*, μ^*), which Γ-split mod B.

Proof. The proof is very similar to that of Lemma 3.2.20. As above, for a string $\alpha \in 2^{<\omega}$ let $F_{\succcurlyeq \alpha} = (F \cap B) \cup \{(\tau, \rho) \in F \ : \ \Gamma(\tau, \rho) \succcurlyeq \alpha\}$, and similarly define $F_{\perp \alpha}$, $E_{\succcurlyeq \alpha}$, $E_{\preccurlyeq \alpha}$ and so on. If $F \cap B$ is (h_1, h_2)-big above (σ^*, μ^*) then we can let $F' = F \cap B$ and $E' = E$. Similarly if $E \cap B$ is (g_1, g_2)-big above (σ, μ).

Suppose otherwise. In that case, for sufficiently long α, $F_{\succcurlyeq \alpha}$ is (h_1, h_2)-small above (σ^*, μ^*). Let α be a string, maximal with respect to $F_{\succcurlyeq \alpha}$ being (h_1, h_2)-big above (σ^*, μ^*). As above we show that either

1. $E_{\perp \alpha}$ is (g_1, g_2)-big above (σ, μ), or
2. $E_{\succcurlyeq \alpha}$ is (g_1, g_2)-big above (σ, μ) and $F_{\perp \alpha}$ is (h_1, h_2)-big above (σ^*, μ^*).

In both cases we can find E' and F' as required.

Again we examine two cases, depending on $E_{\preccurlyeq \alpha}$.

First suppose that $E_{\leqslant\alpha}$ is (g_1, g_2)-big above (σ, μ). Let R witness this. Fix ζ, a leaf of $\operatorname{dom} R$. The argument of the proof of Lemma 3.2.20 is now carried out within $R(\zeta)$. Let $\tau = \zeta^{-\operatorname{dom} A}$. Every $\nu \in E(\zeta)$ extends some unique $\rho \in A(\tau)$. The tree $R(\zeta)$ restricted to initial segments of strings in $A(\tau)$ shows that $A(\tau) \cap R(\zeta)$ is g_2-big above μ; for each $\rho \in A(\tau) \cap R(\zeta)$, $E_{\leqslant\alpha}(\zeta)$ is g_2-big above ρ. The previous argument shows that for each such ρ, $E_{\perp\alpha}(\zeta)$ is g_2-big above ρ. The concatenation property shows that $E_{\perp\alpha}(\zeta)$ is g_2-big above μ. And then $\operatorname{dom} R$ shows that $E_{\perp\alpha}$ is (g_1, g_2)-big above (σ, μ).

Next suppose that $E_{\leqslant\alpha}$ is (g_1, g_2)-small above (σ, μ); the argument is now identical to the comparable one in Lemma 3.2.20, using Lemma 3.3.8. It shows that (2) holds. $\square$

Lemma 3.3.31. *Suppose that $p \in \mathbb{P}_2$ strongly forces that $\Gamma(x^G, y^G)$ is total and forces that $\Gamma(x^G, y^G) \nleqslant_T x^G$.*

Let $C \subset T^p$ be prefix-free and finite; let $g \in \mathcal{Q}$ such that $h^p \gg g$, and $h^p \geqslant 3^{|C|^2} g$ and $g \geqslant b^p$ above $\min\{|\sigma|, |\mu| : (\sigma, \mu) \in C\}$.

Then there is a set $A \subset T^p$, (g, g)-big above C, such that the sets $A \cap (\sigma, \mu^p)^{\leqslant}$ (for $\sigma \in \operatorname{dom} C$) pairwise Γ-split mod B^p.

Proof. We prove the lemma by induction on $|C|$. Let $C^* = C \cup \{(\sigma^* \mu^*)\} \subset T^p$ be finite and prefix-free, and suppose that the lemma is already known for C. Let g satisfy the assumptions of the lemma for C^*. The assumptions of the lemma hold for the set C and the function $3^{|C|} g$. Let A be as guaranteed by the lemma for C and $3^{|C|} g$.

Let $(\sigma_1, \mu_1), (\sigma_2, \mu_2), \ldots, (\sigma_k, \mu_k)$ list the elements of C such that $\sigma_j \neq \sigma^*$. By reverse recursion on $j \leqslant k$ we define a set $A_j \subset T^p$,

$(3^j g, 3^j g)$-big above C^*. We will ensure that $A_j \cap C^{\preccurlyeq} \subset A^{\preccurlyeq}$, and so the sets $A_j \cap (\sigma, \mu^p)^{\preccurlyeq}$ for $\sigma \in \operatorname{dom} C$ pairwise Γ-split mod B^p. Further, we will ensure that $A_{j-1} \cap (\sigma^*, \mu^*)^{\preccurlyeq}$ and $A_{j-1} \cap (\sigma_j, \mu_j)^{\preccurlyeq}$ Γ-split mod B^p; and that $A_{j-1} \subset A_j^{\preccurlyeq}$. Thus in the end, the set A_0 is as required.

We start with $A_k = A \cup \{(\sigma^*, \mu^*)\}$. Now suppose that $j > 0$ and we are given the sets A_j. Let $\tau \in (\operatorname{dom} A_j) \cap \sigma_j^{\preccurlyeq}$. Lemma 3.3.24 says that for all $\tau' \succcurlyeq \tau$ in $\operatorname{dom} T^p$, for all $\rho \in A_j(\tau) \cap \mu_j^{\preccurlyeq}$, the set $\Gamma\text{-Sp}_\rho^{3^j g}(p)$ is $3^j g$-big above σ_j. So applying Lemma 3.3.11 to these sets, and repeating this process for all such τ, we find (finite) $E_{j,0} \subset T^p$ and $E_{j,1} \subset T^p$, uniformly $(3^j g, 3^j g)$-big above $A_j \cap (\sigma_j, \mu_j)^{\preccurlyeq}$, such that for every $(\tau, \rho) \in A_j \cap (\sigma_j, \mu_j)^{\preccurlyeq}$, $E_{j,0} \cap (\tau, \rho)^{\preccurlyeq}$ and $E_{j,1} \cap (\tau, \rho)^{\preccurlyeq}$ locally Γ-split mod B^p. Given $E_j = E_{j,0} \cup E_{j,1}$ we can find $F_j \subset T^p$, $(3^j g, 3^j g)$-big above $A_j \cap (\sigma^*, \mu^*)^{\preccurlyeq}$ (and lying above that set) such that $|\Gamma(\tau, \rho)| > |\Gamma(\tau', \rho')|$ for all $(\tau, \rho) \in F_j \backslash B^p$ and all $(\tau', \rho') \in E_j$. We then appeal to Lemma 3.3.30 with F_j in the role of F, $E_{j,i}$ in the role of E_i, $A_j \cap (\sigma_j, \mu_j)^{\preccurlyeq}$ in the role of A, and using the function $3^{j-1} g$ we get $F_j' \subseteq F_j$, $(3^{j-1} g, 3^{j-1} g)$-big above (σ^*, μ^*) and $E_j' \subseteq E_j$, also $(3^{j-1} g, 3^{j-1} g)$-big above (σ_j, μ_j), which Γ-split mod B^p.

We now define the set A_{j-1}. We first define $\operatorname{dom} A_{j-1}$, and we do this by defining $(\operatorname{dom} A_{j-1}) \cap \sigma^{\preccurlyeq}$ for all $\sigma \in \operatorname{dom} C^*$. Let $\sigma \in \operatorname{dom} C^*$. If $\sigma \neq \sigma_j, \sigma^*$ then $(\operatorname{dom} A_{j-1}) \cap \sigma^{\preccurlyeq} = (\operatorname{dom} A_j) \cap \sigma^{\preccurlyeq}$. We let $(\operatorname{dom} A_{j-1}) \cap (\sigma^*)^{\preccurlyeq} = \operatorname{dom} F_j'$ and $(\operatorname{dom} A_{j-1}) \cap (\sigma_j)^{\preccurlyeq} = \operatorname{dom} E_j'$.

Now for $\tau \in \operatorname{dom} A_{j-1}$ we define $A_{j-1}(\tau)$. Fix such τ; let $\zeta = \tau^{-\operatorname{dom} A_j}$ and let $\sigma = \tau^{-\operatorname{dom} C^*} = \zeta^{-\operatorname{dom} C^*}$. If $\sigma \neq \sigma^*, \sigma_j$ then

$\zeta = \tau$ and we let $A_{j-1}(\tau) = A_j(\tau)$. Otherwise, we define $A_{j-1}(\tau)$ by defining $A_{j-1} \cap \mu^{\preccurlyeq}$ for all $\mu \in C^*(\sigma)$. Suppose that $\sigma = \sigma^*$. If $\mu \neq \mu^*$ then we let $A_{j-1}(\tau) \cap \mu^{\preccurlyeq} = A_j(\zeta) \cap \mu^{\preccurlyeq}$ (which inductively will just equal $A(\tau^{-\operatorname{dom} A}) \cap \mu^{\preccurlyeq}$). We let $A_{j-1}(\tau) \cap (\mu^*)^{\preccurlyeq} = F'_j(\tau)$. Similarly, if $\sigma = \sigma_j$ and $\mu \neq \mu_j$ then we let $A_{j-1}(\tau) \cap \mu^{\preccurlyeq} = A_j(\zeta) \cap \mu^{\preccurlyeq}$; we let $A_{j-1}(\tau) \cap (\mu_j)^{\preccurlyeq} = E'_j(\tau)$. □

Proposition 3.3.32. *Every condition in $\mathbb{P}_2$ forces that if $\Gamma(x^G, y^G)$ is total and $\Gamma(x^G, y^G) \not\leqslant_{\mathrm{T}} x^G$ then $\Gamma(x^G, y^G) \geqslant_{\mathrm{T}} x^G$.*

Proof. The construction is similar to the one in Propositions 3.2.21 and 3.3.28. It is here that we really use the fact that T^p is balanced, for we ensure that each S_k we build is exactly (g, g)-bushy. We assume that $h^p \gg 3^{\bar{g}^2} g$ above $\min\{|\sigma^p|, |\mu^p|\}$ and then apply Lemma 3.3.31 to C being the set of leaves of S_k. We use Lemma 3.1.9. □

And as a result:

Proposition 3.3.33. *Every condition in $\mathbb{P}_2$ forces that* $\deg_{\mathrm{T}}(x^G, y^G)$ *is a strong minimal cover of* $\deg_{\mathrm{T}}(x^G)$.

Remark 3.3.34. We could combine the proofs of Lemmas 3.3.27 and 3.3.31 to build a "totally Γ-splitting" extension: a set A such that if $(\sigma_i, \mu_i) \in C$ (for $i < 2$) and $(\tau_i, \rho_i) \in A \cap (\sigma_i, \mu_i)^{\preccurlyeq} \backslash B$, then $\Gamma(\tau_0, \rho_0) \perp \Gamma(\tau_1, \rho_1)$ provided that either $\sigma_0 \neq \sigma_1$, or $\tau_0 = \tau_1$ (and $\rho_0 \neq \rho_1$). We could then have a single construction (replacing Propositions 3.3.32 and 3.3.33) giving a condition forcing that $\Gamma(x^G, y^G) \equiv_{\mathrm{T}} (x^G, y^G)$.

3.4 The general step

We now generalise to get a linearly ordered initial segment of length n. Once the correct definitions are in place, much of the development closely follows the previous section.

3.4.1 Length *n* forest systems

We work with n-tuples of strings. We use boldface notation for tuples. If $\boldsymbol{\tau}$ is a tuple then τ_i denotes the i^{th} component of $\boldsymbol{\tau}$. The partial ordering of extension $\preccurlyeq$ on $(\omega^{<\omega})^n$ is defined as expected. For a set $A \subseteq (\omega^{<\omega})^n$ we let $A^{\preccurlyeq}$ be the upward closure of A under this partial ordering. If $\boldsymbol{\tau}$ is an n-tuple and $k \leqslant n$ then we let $\boldsymbol{\tau}\restriction_k = (\tau_1, \dots, \tau_k)$ and $\boldsymbol{\tau}\restriction_{(k,n]} = (\tau_{k+1}, \dots, \tau_n)$.

For a set $A \subseteq (\omega^{<\omega})^n$ and $k < n$ we let $\operatorname{dom}_k A$ be the domain of A thought of as a relation between k-tuples and $(n-k)$-tuples:

$$\operatorname{dom}_k A = \{\boldsymbol{\tau}\restriction_k : \boldsymbol{\tau} \in A\}.$$

For $\boldsymbol{\tau} \in (\omega^{<\omega})^k$ we let

$$A(\boldsymbol{\tau}) = \left\{\boldsymbol{\rho} \in (\omega^{<\omega})^{n-k} \ : \ (\boldsymbol{\tau}, \boldsymbol{\rho}) \in A\right\}.$$

We will frequently need to chop off the last bit, so for compact notation we let $\boldsymbol{\tau}{\downarrow} = \boldsymbol{\tau}\restriction_{n-1}$ for all $\boldsymbol{\tau} \in (\omega^{<\omega})^n$, and let $A{\downarrow} = \operatorname{dom}_{n-1} A = \{\boldsymbol{\tau}{\downarrow} : \boldsymbol{\tau} \in A\}$ for all $A \subseteq (\omega^{<\omega})^n$.

Definition 3.4.1. By induction on n we define the notion of a *prefix-free* set of tuples of strings: a set $A \subset (\omega^{<\omega})^n$ is prefix-free if $A{\downarrow}$ is prefix-free, and for all $\boldsymbol{\tau} \in A{\downarrow}$, $A(\boldsymbol{\tau})$ is a prefix-free set of strings.

If A is prefix-free and $\boldsymbol{\tau} \in A^{\preccurlyeq}$ then there is a unique $\boldsymbol{\sigma} \in A$ such that $\boldsymbol{\sigma} \preccurlyeq \boldsymbol{\tau}$ (formally this is proved by induction on n); we denote this $\boldsymbol{\sigma}$ by $\boldsymbol{\tau}^{-A}$. Note that if A is prefix-free and $\boldsymbol{\tau} \in A^{\preccurlyeq}$ then $\boldsymbol{\tau}{\downarrow} \in (A{\downarrow})^{\preccurlyeq}$ and $(\boldsymbol{\tau}{\downarrow})^{-A{\downarrow}} = \boldsymbol{\tau}^{-A}{\downarrow}$.

Definition 3.4.2. By induction on n we define the notion of a length n forest system. Let $A \subset (\omega^{<\omega})^n$ be prefix-free and finite. A *length n forest system above A* is a set $T \subseteq A^{\preccurlyeq}$ such that:

- $T{\downarrow}$ is a length $n-1$ forest system above $A{\downarrow}$;
- for all $\boldsymbol{\tau} \in T{\downarrow}$, $T(\boldsymbol{\tau})$ is a finite forest above $A(\boldsymbol{\tau}^{-A{\downarrow}})$;
- if $\boldsymbol{\tau} \preccurlyeq \boldsymbol{\tau}' \in T{\downarrow}$ then $T(\boldsymbol{\tau}')$ is an end-extension of $T(\boldsymbol{\tau})$.

A forest system S is a subsystem of T if $S \subseteq T$. We write $\ell(T)$ for the length of T. If A is a singleton $\boldsymbol{\sigma}$ then we say that T is a *tree system* above $\boldsymbol{\sigma}$.

Lemma 3.4.3. *Let T be a tree system and let $\boldsymbol{\sigma} \in T$. Then $T \cap \boldsymbol{\sigma}^{\preccurlyeq}$ is a tree system above σ.*

(In fact σ can be replaced by any finite, prefix-free subset of T).

Proof. By induction on $\ell(T)$. Let $R = T \cap \sigma^{\preccurlyeq}$. The point is that $R{\downarrow} = T{\downarrow} \cap (\boldsymbol{\sigma}{\downarrow})^{\preccurlyeq}$. For suppose that $\boldsymbol{\tau} \in T{\downarrow} \cap (\boldsymbol{\sigma}{\downarrow})^{\preccurlyeq}$. Then $T(\boldsymbol{\sigma}{\downarrow}) \subseteq T(\boldsymbol{\tau})$ and $\boldsymbol{\sigma} \in T$ imply that $(\boldsymbol{\tau}, \sigma_n) \in T$ and witnesses that $\boldsymbol{\tau} \in R{\downarrow}$. Finally we also observe that for $\boldsymbol{\tau} \in R{\downarrow}$ we have $R(\boldsymbol{\tau}) = T(\boldsymbol{\tau}) \cap (\sigma_n)^{\preccurlyeq}$. □

The definition of an h-bounded (and so of a computably bounded) tree system is as expected. If T is computable and computably bounded then for all $k < \ell(T)$, $\mathrm{dom}_k T$ is computable and the map $\boldsymbol{\tau} \mapsto T(\boldsymbol{\tau})$ is computable.

A *leaf* of a forest system T is a $\preccurlyeq$-maximal element of T. A tuple $\boldsymbol{\tau}$ is a leaf of T if and only if $\boldsymbol{\tau}{\downarrow}$ is a leaf of $T{\downarrow}$ and $\tau_{\ell(T)}$ is a leaf of $T(\boldsymbol{\tau}{\downarrow})$. The set of leaves of a forest system is prefix-free.

If T and S are length n forest systems then we say that T is an *end-extension* of S if:

- $T{\downarrow}$ is an end-extension of $S{\downarrow}$;
- If $\boldsymbol{\tau} \in S{\downarrow}$ is not a leaf of $S{\downarrow}$ then $T(\boldsymbol{\tau}) = S(\boldsymbol{\tau})$;
- If $\boldsymbol{\tau}$ is a leaf of $S{\downarrow}$ then $T(\boldsymbol{\tau})$ is an end-extension of $S(\boldsymbol{\tau})$.

Note that this is a transitive relation.

Lemma 3.4.4. *Let $\langle S_m \rangle$ be a sequence of forest systems above A, with each S_{m+1} an end-extension of S_m. Then $\bigcup_m S_m$ is a forest system above A.*

Proof. Let $S = \bigcup_m S_m$. Then $S{\downarrow} = \bigcup_m S_m{\downarrow}$, and so by induction on the length, $S{\downarrow}$ is a forest system above $A{\downarrow}$. Let $\boldsymbol{\tau} \in S{\downarrow}$. Then $S(\boldsymbol{\tau}) = \bigcup_m S_m(\boldsymbol{\tau})$ is the union of a sequence of end-extensions above $A(\boldsymbol{\tau}^{-A{\downarrow}})$, and so is a forest above that set; note that if $\boldsymbol{\tau} \in S_m{\downarrow}$ but is not a leaf of $S_m{\downarrow}$ then $S(\boldsymbol{\tau}) = S_m(\boldsymbol{\tau})$. $\square$

Other breaking points

We don't have to isolate only the last coordinate. For example:

Lemma 3.4.5. *Let $A \subseteq (\omega^{<\omega})^n$. The following are equivalent:*

1. *A is prefix-free;*
2. *For some $k \in \{1, \dots, n-1\}$, $\mathrm{dom}_k A$ is prefix-free and for all $\boldsymbol{\tau} \in \mathrm{dom}_k A$, $A(\boldsymbol{\tau})$ is prefix-free; and*
3. *For all $k \in \{1, \dots, n-1\}$, $\mathrm{dom}_k A$ is prefix-free and for all $\boldsymbol{\tau} \in \mathrm{dom}_k A$, $A(\boldsymbol{\tau})$ is prefix-free.*

The proof relies on the fact that $(A{\downarrow})(\boldsymbol{\tau}) = (A(\boldsymbol{\tau})){\downarrow}$, and induction. For forest systems we do not get as nice a result.

Lemma 3.4.6. *Let $A \subset (\omega^{<\omega})^n$ be prefix-free and let $T \subseteq A^{\preccurlyeq}$.*

1. *Suppose that T is a forest system above A. Then for all $k \in \{1, 2, \dots, n-1\}$: (a) $\mathrm{dom}_k T$ is a forest system above $\mathrm{dom}_k A$; (b) For all $\boldsymbol{\tau} \in \mathrm{dom}_k T$, $T(\boldsymbol{\tau})$ is a forest system above $A(\boldsymbol{\tau}^{-\,\mathrm{dom}_k A})$; and (c) if $\boldsymbol{\tau} \preccurlyeq \boldsymbol{\tau}'$ are in $\mathrm{dom}_k T$ then $T(\boldsymbol{\tau}) \subseteq T(\boldsymbol{\tau}')$.*
2. *Let $k \in \{1, 2, \dots, n-1\}$; suppose that $\mathrm{dom}_k T$ is a forest system above $\mathrm{dom}_k A$, that for all $\boldsymbol{\tau} \in \mathrm{dom}_k T$, $T(\boldsymbol{\tau})$ is a forest system above $A(\boldsymbol{\tau}^{-\,\mathrm{dom}_k A})$, and that if $\boldsymbol{\tau} \preccurlyeq \boldsymbol{\tau}'$ are in $\mathrm{dom}_k T$ then $T(\boldsymbol{\tau})$ is an end-extension of $T(\boldsymbol{\tau}')$. Then T is a forest system above A.*

Again the proof is routine. In the situation of (1) we don't always get that $T(\boldsymbol{\tau}')$ end-extends $T(\boldsymbol{\tau})$. Suppose for example that $\tau \prec \tau'$ are in $\mathrm{dom}_1 T$ and that $\rho \prec \rho'$ are in $\mathrm{dom}_1 T(\tau)$ (and so also in $\mathrm{dom}_1 T(\tau')$). It is possible that $T(\tau', \rho) \neq T(\tau, \rho)$, even though ρ is not a leaf of $T(\tau)$. For example we could have $T(\tau', \rho') =$

$T(\tau', \rho) = T(\tau, \rho')$ which is a proper end-extension of $T(\tau, \rho)$. For end-extending, though, we do get full invariance of breaking point:

Lemma 3.4.7. *Let S and T be forest systems of length n. The following are equivalent:*

1. *T is an end-extension of S;*
2. *For some $k \in \{1, \ldots, n-1\}$, $\mathrm{dom}_k T$ is an end-extension of $\mathrm{dom}_k S$, for all $\boldsymbol{\tau} \in \mathrm{dom}_k S$, $T(\boldsymbol{\tau})$ is an end-extension of $T(\boldsymbol{\tau})$, and if $\boldsymbol{\tau} \in \mathrm{dom}_k S$ is not a leaf of $\mathrm{dom}_k S$, then $T(\boldsymbol{\tau}) = S(\boldsymbol{\tau})$.*
3. *For all $k \in \{1, \ldots, n-1\}$, $\mathrm{dom}_k T$ is an end-extension of $\mathrm{dom}_k S$, for all $\boldsymbol{\tau} \in \mathrm{dom}_k S$, $T(\boldsymbol{\tau})$ is an end-extension of $T(\boldsymbol{\tau})$, and if $\boldsymbol{\tau} \in \mathrm{dom}_k S$ is not a leaf of $\mathrm{dom}_k S$, then $T(\boldsymbol{\tau}) = S(\boldsymbol{\tau})$.*

Also note that if S is a forest system then $\boldsymbol{\tau} \in S$ is a leaf of S if and only if for some (all) $k \in \{1, 2, \ldots, \ell(S) - 1\}$, $\restriction\restriction k$ is a leaf of $\mathrm{dom}_k S$ and $\boldsymbol{\tau}\restriction_{(k, \ell(S))]}$ is a leaf of $S(\boldsymbol{\tau}\restriction_k)$.

Paths of tree systems

We simplify our presentation by restricting ourselves to balanced tree systems.

Definition 3.4.8. Let T be a tree system and let $m < \omega$. We say that m is a *balanced level of T* if for all $\tau \in \mathrm{dom}_1 T$ of length m, every component of every leaf of $T(\tau)$ has length m. We say that T is *balanced* if $\mathrm{dom}_1 T$ has no leaves and T has infinitely many balanced levels.

For a balanced tree system T we let

$$[T] = \left\{ \boldsymbol{x} \in (\omega^\omega)^{\ell(T)} \ : \ \boldsymbol{x}\restriction_m \in T \text{ for every balanced level } m \text{ of } T \right\}.$$

The set $[T]$ is a closed subset of $(\omega^\omega)^n$.

For $\boldsymbol{x} \in [T\downarrow]$ we let $T(\boldsymbol{x}) = \bigcup_{\boldsymbol{\tau}<\boldsymbol{x}} T(\boldsymbol{\tau})$. This is a tree with no leaves. If T is balanced then so is $T\downarrow$, and $[T] = \{(\boldsymbol{x}, y) \ : \ \boldsymbol{x} \in [T\downarrow] \ \& \ y \in [T(\boldsymbol{x})]\}$. If T is balanced, computable and computably bounded then $[T]$ is effectively closed.

Bushiness for forest systems

Let $\boldsymbol{g} = (g_1, \dots, g_n)$ be a tuple of bounding functions, and let T be a length n forest system. We say that T is $\boldsymbol{g}$-bushy if $T\downarrow$ is $\boldsymbol{g}\downarrow$-bushy and for all $\boldsymbol{\tau} \in T\downarrow$, $T(\boldsymbol{\tau})$ is g_n-buhsy. As usual, T is $\boldsymbol{g}$-bushy if and only if for some (all) $k \in \{1, 2, \dots, n-1\}$, $\operatorname{dom}_k T$ is $\boldsymbol{g}\restriction_k$-bushy and for all $\boldsymbol{\tau} \in \operatorname{dom}_k T$, $T(\boldsymbol{\tau})$ is $\boldsymbol{g}\restriction_{(k,n]}$-bushy.

We say that a set $B \subseteq (\omega^{<\omega})^n$ is $\boldsymbol{g}$-*big* above some finite prefix-free set $A \subset (\omega^{<\omega})^n$ if there is a $\boldsymbol{g}$-bushy finite forest system R above A whose leaves lie in B. This is extended to all sets A as above. For $k < n$, $B \subseteq (\omega^{<\omega})^n$, a finite, prefix-free $A \subseteq (\omega^{<\omega})^n$ and an $(n-k)$-tuple $\boldsymbol{h}$ of bounding functions we let

$$\pi_A^{\boldsymbol{h}}(B) = \left\{ \boldsymbol{\tau} \in (\operatorname{dom}_k A)^{\preccurlyeq} \ : \ B(\boldsymbol{\tau}) \text{ is } \boldsymbol{h}\text{-big above } A(\boldsymbol{\tau}^{-\operatorname{dom}_k A}) \right\}.$$

Note that this notation is different from the one used in the previous section; however, if A is a singleton $\boldsymbol{\sigma}$ then we revert to the old notation and write $\pi^{\boldsymbol{h}}_{\boldsymbol{\sigma}\restriction_{(k,n]}}(B)$ instead of $\pi^{\boldsymbol{h}}_{\boldsymbol{\sigma}}(B)$. A set B is $\boldsymbol{g}$-big above A

if and only $\pi_A^{\boldsymbol{g}\upharpoonright(k,n]}(B)$ is $\boldsymbol{g}\upharpoonright_k$-big above A. The proof of this follows the proof of Lemma 3.3.7, using Lemma 3.4.6(2) (and the fact that every finite prefix-free set is a forest system above itself, and any forest system R above A is an end-extension of A). The proof gives the analogue of Remark 3.3.14: if B is $\boldsymbol{g}$-big above A, T is a forest system and $A, B \subseteq T$ then a finite forest system S witnessing the largeness can be taken to be a subset of T.

Remark 3.4.9. Let $1 \leqslant k < m < n$, let $\boldsymbol{\sigma} \in (\omega^{<\omega})^{m-k}$, $\boldsymbol{\mu} \in (\omega^{<\omega})^{n-m}$, $\boldsymbol{g}$ be an $(m-k)$-tuple of bounding functions, and an $(n-m)$-tuple of bounding functions. Let $B \subseteq (\omega^{<\omega})^n$. Then

$$\pi_{\boldsymbol{\sigma}}^{\boldsymbol{g}}(\pi_{\boldsymbol{\mu}}^{\boldsymbol{h}}(B)) = \pi_{\boldsymbol{\sigma},\boldsymbol{\mu}}^{\boldsymbol{g},\boldsymbol{h}}(B).$$

The big subset property holds for largeness over singletons, with the same proof as that of Lemma 3.3.8.

For the weak concatenation property, we will straightaway work within tree systems. But first we discuss concatenations. Suppose that S is a finite forest system, that A is the set of leaves of S, and that R is a forest system above A. Since S is finite, $A\!\downarrow$ is the set of leaves of $S\!\downarrow$. We then define $S\hat{\ }R$ by letting:

- $(S\hat{\ }R)\!\downarrow = S\!\downarrow\hat{\ }R\!\downarrow$;
- For $\boldsymbol{\tau} \in S\!\downarrow$, not a leaf of $S\!\downarrow$, we let $(S\hat{\ }R)(\boldsymbol{\tau}) = S(\boldsymbol{\tau})$;
- For $\boldsymbol{\tau} \in R\!\downarrow$ we let $(S\hat{\ }R)(\boldsymbol{\tau}) = S(\boldsymbol{\tau}^{-A})\hat{\ }R(\boldsymbol{\tau}) = S(\boldsymbol{\tau}^{-A}) \cup R(\boldsymbol{\tau})$.

Then $S\hat{\ }R$ is an end-extension of S, whose leaves are the leaves of R. Also note that if $S, R \subseteq T$ for some forest system T then $S\hat{\ }R \subseteq T$.

If both S and R are $\boldsymbol{g}$-bushy then so is $S\hat{\ }R$. We thus get the restricted analogue of Lemma 3.3.10. From now we fix a forest system T.

- Suppose that B is $\boldsymbol{g}$-big above A, and that C is $\boldsymbol{g}$-big above B. Then C is $\boldsymbol{g}$-big above A. If $A, B, C \subseteq T$ then every forest system $S \subseteq T$ witnessing that B is $\boldsymbol{g}$-big above A has an end-extension $R \subseteq T$ which witnesses that C is $\boldsymbol{g}$-big above A.

We get an analogue of Lemma 3.3.11. The notion of an open subset of T is as expected.

Lemma 3.4.10. *Let $\mathcal{B}$ be a finite family of subsets of T which are open in T. Let $A \subseteq T$ be finite and prefix-free. Suppose that each $B \in \mathcal{B}$ is $\boldsymbol{g}$-big above $A^{\preccurlyeq} \cap T$ (recall that this means that it is $\boldsymbol{g}$-big above every finite, prefix-free subset of $A^{\preccurlyeq} \cap T$). Then $\bigcap \mathcal{B}$ is $\boldsymbol{g}$-big above A.*

We can now prove the analogue of Lemma 3.3.12.

Lemma 3.4.11. *Let T be a forest system and let $A, B \subseteq T$; suppose that B is open in T. Suppose that for all $\boldsymbol{\tau} \in A^{\preccurlyeq} \cap T$, B is $\boldsymbol{g}$-big above $\boldsymbol{\tau}$. Then B is $\boldsymbol{g}$-big above A.*

Proof. By induction on the length of T. We may assume that A is finite and prefix-free. We need to show that $C = \pi_A^{g_n}(B)$ is $\boldsymbol{g}\!\downarrow$-big above $A\!\downarrow$. Let $\boldsymbol{\tau} \in (A\!\downarrow)^{\preccurlyeq} \cap T\!\downarrow$. We claim that C is $\boldsymbol{g}\!\downarrow$-big above $\boldsymbol{\tau}$ (and then apply the induction hypothesis). Let $\boldsymbol{\sigma} = \boldsymbol{\tau}^{-A\downarrow}$. Then $C \cap \boldsymbol{\sigma}^{\preccurlyeq}$ equals $\bigcap_{\mu \in A(\boldsymbol{\sigma})} \pi_\mu^{g_n}(B)$. By assumption, each $\pi_\mu^{g_n}(B)$ is $\boldsymbol{g}\!\downarrow$-big above every tuple in $\boldsymbol{\sigma}^{\preccurlyeq} \cap T\!\downarrow$; we apply the analogue of Lemma 3.3.11 mentioned above. ◻

Corollary 3.4.12. *Let T be a tree system, let $A, B, C \subseteq T$, and suppose that C is open in T. Suppose that B is $\boldsymbol{g}$-big above A, and that C is $\boldsymbol{g}$-big above every tuple in $B^{\preccurlyeq} \cap T$. Then C is $\boldsymbol{g}$-big above A, and in fact every finite $\boldsymbol{g}$-bushy forest system $S \subseteq T$ which witnesses that B is $\boldsymbol{g}$-big above A has an end-extension $R \subseteq T$ which witnesses that C is $\boldsymbol{g}$-big above A.*

As a corollary we get the analogue of Lemma 3.3.17:

- If T is a bounded and balanced $\boldsymbol{b}$-bushy tree system above $\boldsymbol{\sigma}$, and $B \subset T$ is open in T and $\boldsymbol{b}$-small above $\boldsymbol{\sigma}$, then for every m there is some $\boldsymbol{\tau} \in T$ such that $|\tau_i| \geqslant m$ for all $i \leqslant \ell(T)$, and above which B is $\boldsymbol{b}$-small.

3.4.2 The notion of forcing and restriction maps

We let B_{DNC^n} be the set of tuples $\boldsymbol{\tau} \in (\omega^{<\omega})^n$ such that either $\boldsymbol{\tau}{\downarrow} \in B_{\mathrm{DNC}^{n-1}}$, or $\tau_n \in B_{\mathrm{DNC}^{\boldsymbol{\tau}\downarrow}}$, that is, if there is some $e < |\tau_n|$ such that $\tau_n(e) = J^{\boldsymbol{\tau}\downarrow}(e)$.

For brevity, for a tuple $\boldsymbol{\sigma} \in (\omega^{<\omega})^n$ we let $|\boldsymbol{\sigma}| = \min\{|\sigma_i| : i \leqslant n\}$. When a tuple-length n is clear from the context, then for a function g we let $\boldsymbol{g} = (g, g, \ldots, g)$.

We let $\mathbb{P}_n$ be the set of tuples $p = (\boldsymbol{\sigma}^p, T^p, B^p, h^p, b^p)$ satisfying:

1. T^p is a computably bounded, computable, balanced tree system above $\boldsymbol{\sigma}^p$;

2. $h^p \in Q$ and T^p is $\boldsymbol{h}^p$-bushy;

3. $B^p \subset T^p$ is c.e. and open in T^p, and $B^p \supseteq B_{\mathrm{DNC}^n} \cap T^p$;

4. $b^p \in Q$ and B^p is $\boldsymbol{b}^p$-small above $\boldsymbol{\sigma}^p$; and

5. $h^p \gg b^p$ and $h^p \geqslant b^p$ above $|\boldsymbol{\sigma}^p|$.

We define a partial ordering on $\mathbb{P}_n$ as follows. A condition q extends a condition p if $\boldsymbol{\sigma}^p \leqslant \boldsymbol{\sigma}^q$, T^q is a subsystem of T^p, $B^p \cap T^q \subseteq B^q$, and $h^q \leqslant h^p$ and $b^q \geqslant b^p$ above $|\boldsymbol{\sigma}^q|$.

The assignment of closed sets $X^p = [T^p] \backslash [B^p]^{\prec}$ for $p \in \mathbb{P}_n$ is acceptable; the proof is identical to the proof of Lemma 3.3.19.

If $G \subset \mathbb{P}_n$ is sufficiently generic then we denote the generic tuple (the element of the singleton $\bigcap_{p \in G}[T^p] \backslash [B^p]^{\prec}$) by $\boldsymbol{x}^G$. As above, every condition in $\mathbb{P}_n$ forces that x_n^G is DNC relative to $\boldsymbol{x}^G\downarrow$.

The restriction maps

For all $n \geqslant 2$, define $i_n \colon \mathbb{P}_n \to \mathbb{P}_{n-1}$ by letting

$$i_n(q) = \left(\boldsymbol{\sigma}^q\downarrow, T^q\downarrow, \pi^{b^q}_{\sigma^q_n}(B^q), h^q, b^q\right),$$

where we have

$$\pi^{b^q}_{\sigma^q_n}(B^q) = \{\boldsymbol{\tau} \in T^q\downarrow \colon\ B^q(\boldsymbol{\tau}) \text{ is } b^q\text{-big above } \sigma^q_n\}\,.$$

It is routine to check that $i_n(q) \in \mathbb{P}_{n-1}$ for all $q \in \mathbb{P}_n$. Inductively we define $\mathbb{Q}_n \subset \mathbb{P}_n$: $\mathbb{Q}_1 = \mathbb{P}_1$, and $\mathbb{Q}_n$ is the set of conditions $q \in \mathbb{Q}_n$ such that:

- $i_n(q) \in \mathbb{Q}_{n-1}$; and

- $\pi^{b^q}_{\sigma^q}(B^q) = \{\boldsymbol{\tau} \in T^q\!\downarrow : \sigma^q_n \in B^q(\boldsymbol{\tau})\}$.

We again observe that for all $q \in \mathbb{Q}_n$, $X^q\!\downarrow = X^{i_n(q)}$; the proof is the same as above. The proof that the restriction of i_n to $\mathbb{Q}_n$ is order-preserving is identical to that in the proof of Proposition 3.3.21.

Lemma 3.4.13. *There is a map $\nu_n \colon \mathbb{P}_n \to \mathbb{Q}_n$ such that:*

1. *$\nu_n(q) \leqslant q$ for all $q \in \mathbb{P}_n$; and*
2. *$i_n \circ \nu_n = \nu_{n-1} \circ i_n$.*

In particular, $\mathbb{Q}_n$ is dense in $\mathbb{P}_n$.

Proof. We omit the indices n and $n-1$ from i_n, ν_n etc.; they will be clear from the context.

Let $q \in \mathbb{P}_n$. For brevity we let $C_n = B^q$ and for $k \in \{1, \dots, n-1\}$ we let $C_k = \pi^{\boldsymbol{b}^q}_{\boldsymbol{\sigma}^q \upharpoonright (k,n]}(B^q)$. Remark 3.4.9 says that if $k < m \leqslant n$ then $C_k = \pi^{\boldsymbol{b}^q}_{\boldsymbol{\sigma}^q \upharpoonright (k,m]}(C_m)$.

We define a tuple $\nu(q) = (\boldsymbol{\sigma}^q, T^q, B^{\nu(q)}, h^q, b^q)$ by letting

$$B^{\nu(q)} = \{\boldsymbol{\tau} \in T^q \ : \ \boldsymbol{\tau}\upharpoonright_k \in C_k \text{ for some } k \leqslant n\}\,.$$

The set $B^{\nu(q)}$ is $\boldsymbol{b}^q$-small above $\boldsymbol{\sigma}^q$. For let D be the set of leaves of a $\boldsymbol{b}^q$-bushy finite tree system $S \subset T^q$ above $\boldsymbol{\sigma}^q$. Since C_1 is b^q-small above σ^q_1 we find some $\tau_1 \in (\mathrm{dom}_1 D)\backslash C_1$. Since $C_1 = \pi^{b^q}_{\sigma^q_2}(C_2)$, $C_2(\tau)$ is b^q-small above σ^q_2; we find some τ_2 such that $(\tau_1, \tau_2) \in (\mathrm{dom}_2 D)\backslash C_2$; and so on, we find some $\boldsymbol{\tau} \in D\backslash B^{\nu(q)}$. We conclude that $\nu(q) \in \mathbb{P}_n$ (and $\nu(q) \leqslant q$).

Now $B^{i(q)} = C_{n-1}$; so $B^{\nu(i(q))}$ is the set of tuples $\boldsymbol{\tau} \in T^q{\downarrow}$ such that $\boldsymbol{\tau}\restriction_k \in C_k$ for some $k \leqslant n-1$.

Let $\boldsymbol{\tau} \in T^q$. If $\boldsymbol{\tau}{\downarrow} \in B^{\nu(i(q))}$ then $B^{\nu(q)}(\boldsymbol{\tau}) = T^q(\boldsymbol{\tau})$, in particular $\sigma_n^q \in B^{\nu(q)}(\boldsymbol{\tau})$. Otherwise, $B^{\nu(q)}(\boldsymbol{\tau}) = B^q(\tau)$, and since in this case $\boldsymbol{\tau} \notin C_{n-1}$ we see that $B^{\nu(q)}(\boldsymbol{\tau})$ is b^q-small above σ_n^q. We conclude that $B^{i(\nu(q))} = \pi^{b^q}_{\sigma_n^q}(B^{\nu(q)}) = B^{\nu(i(q))}$ and so that $i(\nu(q)) = \nu(i(q))$.

We also conclude that $\boldsymbol{\tau} \in \pi^{b^q}_{\sigma_n^q}(B^{\nu(q)})$ if and only if $\sigma_n^q \in B^{\nu(q)}(\boldsymbol{\tau})$. By induction, $\nu(i(q)) \in \mathbb{Q}_{n-1}$, so $\nu(q) \in \mathbb{Q}_n$. □

Proposition 3.4.14. *$i_n\restriction_{\mathbb{Q}_n}$ is a restriction map from $\mathbb{Q}_n$ to $\mathbb{Q}_{n-1}$.*

Proof. It remains to show that if $q \in \mathbb{Q}_n$ and $p \in \mathbb{Q}_{n-1}$ extends $i_n(q)$ then there is some $r \in \mathbb{Q}_n$ extending q such that $i_n(r) \leqslant p$. By using the map ν_n, it suffices to find $r \in \mathbb{P}_n$. The proof is identical to that of Proposition 3.3.21. □

Lemma 3.4.15. *$i_n\restriction_{\mathbb{Q}_n}$ is onto $\mathbb{Q}_{n-1}$.*

Proof. Let $p \in \mathbb{Q}_{n-1}$. We define $q \in \mathbb{Q}_n$ such that $i_n(q) = p$ by letting, for $\boldsymbol{\sigma} \in T^p$, $T^q(\boldsymbol{\sigma}) = (h^p)^{\leqslant|\sigma|}$, and let $B^q(\boldsymbol{\sigma}) = T^q(\boldsymbol{\sigma})$ if $\boldsymbol{\sigma} \in B^p$, otherwise $B^q(\boldsymbol{\sigma}) = B_{\mathrm{DNC}^{\sigma}}$. □

Totality

Proposition 3.4.16. *Let $C \subseteq (\omega^\omega)^n$ be Π^0_2 and let $p \in \mathbb{P}_n$. If $p \Vdash \boldsymbol{x}^G \in C$ then p has an extension which strongly forces that $\boldsymbol{x}^G \in C$.*

The proof is identical to the proof of Proposition 3.3.23.

3.4.3 Minimality

Let $\Gamma\colon (\omega^\omega)^n \to 2^\omega$ be a Turing functional.

Definition 3.4.17. Let $B \subseteq (\omega^{<\omega})^n$. Two sets $A_0, A_1 \subset (\omega^{<\omega})^n$ form a *local* Γ*-splitting mod* B if for all $\boldsymbol{\tau} \in (\omega^{<\omega})^{n-1}$, the sets $A_0(\boldsymbol{\tau})$ and $A_1(\boldsymbol{\tau})$ $\Gamma(\boldsymbol{\tau}, -)$-split mod $B(\boldsymbol{\tau})$.

Definition 3.4.18. Let $A \subset (\omega^{<\omega})^n$ be finite and prefix-free, and let $\mathcal{B}$ be a collection of subsets of $(\omega^{<\omega})^n$. We say that the sets in $\mathcal{B}$ are *uniformly* $\boldsymbol{g}$*-big above* A if $\bigcap_{B\in\mathcal{B}} \pi_A^{g_n}(B)$ is $\boldsymbol{g}{\downarrow}$-big above $A{\downarrow}$.

Lemma 3.4.19. *Suppose that* $p \in \mathbb{P}_n$ *strongly forces that* $\Gamma(\boldsymbol{x}^G)$ *is total, and forces that it is not computable from* $\boldsymbol{x}^G{\downarrow}$*. Let* $\boldsymbol{\sigma} \in T^p$*; let* $g \in \mathcal{Q}$ *such that* $h^p \gg g$*, and* $h^p \geqslant 3g$ *and* $g \geqslant b^p$ *above* $|\boldsymbol{\sigma}|$*. Then there are sets* $A_0, A_1 \subset T^p$*, uniformly* $\boldsymbol{g}$*-big above* $\boldsymbol{\sigma}$*, which locally* Γ*-split mod* B^p*.*

Proof. Identical to the proof of Lemma 3.3.24. □

Lemma 3.4.20. *Let* $\boldsymbol{g}$ *and* $\boldsymbol{h}$ *be n-tuples of bounding functions; let* $B \subseteq (\omega^{<\omega})^n$ *be open. Suppose that:*

- $\boldsymbol{\sigma}, \boldsymbol{\sigma}^* \in (\omega^{<\omega})^n$*;*
- A *is* $3\boldsymbol{g}$*-big above* $\boldsymbol{\sigma}$*;*
- E_0 *and* E_1 *are uniformly* $3\boldsymbol{g}$*-big above* A*; and for all* $\boldsymbol{\tau} \in A$*,* $E_0 \cap \boldsymbol{\tau}^{\preccurlyeq}$ *and* $E_1 \cap \boldsymbol{\tau}^{\preccurlyeq}$ *locally* Γ*-split mod* B*; and*
- F *is* $3\boldsymbol{h}$*-big above* $\boldsymbol{\sigma}^*$*, and* $|\Gamma(\boldsymbol{\rho})| > |\Gamma(\boldsymbol{\zeta})|$ *for all* $\boldsymbol{\rho} \in F\backslash B$ *and all* $\boldsymbol{\zeta} \in E\backslash B$*, where* $E = E_0 \cup E_1$*.*

Then there are $E' \subseteq E$, $\boldsymbol{g}$-big above $\boldsymbol{\sigma}$, and $F' \subseteq F$, $\boldsymbol{h}$-big above $\boldsymbol{\sigma}^*$, which Γ-split mod B.

Proof. Identical to the proof of Lemma 3.3.30. □

Lemma 3.4.21. *Suppose that $p \in \mathbb{P}_n$ strongly forces that $\Gamma(\boldsymbol{x}^G)$ is total, and forces that it is not computable from $\boldsymbol{x}^G\downarrow$. Let $k \in \{0, 1, \dots, n-1\}$. Let $C \subset T^p$ be finite and prefix-free. Let $g \in \mathcal{Q}$ such that $h^p \gg g$, and $h^p \geqslant 3^{|C|^2} g$ and $g \geqslant b^p$ above $|\boldsymbol{\sigma}|$ for all $\boldsymbol{\sigma} \in C$.*

Then there is a set $A \subset T^p$, $\boldsymbol{g}$-big above C, such that for all $\boldsymbol{\tau} \in \mathrm{dom}_k A$, the sets in the collection

$$\left\{A(\boldsymbol{\tau}) \cap (\rho, \boldsymbol{\sigma}^p\restriction_{(k+1,n]})^{\leqslant} \,:\, \rho \in \mathrm{dom}_1 A(\boldsymbol{\tau})\right\}$$

pairwise $\Gamma(\boldsymbol{\tau}, -)$-split mod $B^p(\boldsymbol{\tau})$.

Proof. The notation for the case $k = 0$ is slightly easier. In this case we closely follow the proof of Lemma 3.3.31. For simplicity of notation, for a set $A \subseteq T^p$ and some tuple $\boldsymbol{\tau} \in \mathrm{dom}_k T^p$ (for some $k < n$) we let $A \cap (\boldsymbol{\tau})^{\leqslant} = A \cap (\boldsymbol{\tau}, \boldsymbol{\sigma}^p\restriction_{(k,n]})^{\leqslant}$. We prove the lemma by induction on $|C|$; we let $C^* = C \cup \{\boldsymbol{\sigma}^*\}$; by induction we are given A which is $3^{|C|}g$-big above C, and the sets $A \cap (\rho)^{\leqslant}$ (for $\rho \in \mathrm{dom}_1 C$) pairwise Γ-split mod B^p. We list the elements $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2, \dots, \boldsymbol{\sigma}_m$ of C such that $(\sigma_j)_1 \neq \sigma^*_1$. By reverse recursion on $j \leqslant m$ we define sets $A_j \subset T^p$ with $A_{j-1} \subset A_j^{\leqslant}$ and $A_m \cap \boldsymbol{\sigma}^{\leqslant} \subset A^{\leqslant}$ for all $\boldsymbol{\sigma} \in C$. We ensure that A_j is $3^j\boldsymbol{g}$-big above C^* and that $A_{j-1} \cap \boldsymbol{\sigma}_j^{\leqslant}$ and $A_{j-1} \cap \boldsymbol{\sigma}^*$ form a Γ-splitting mod B.

We start with $A_m = A \cup \{\boldsymbol{\sigma}^*\}$. Say we are given A_j, $j > 0$. For brevity let $D_j = (A_j \cap \boldsymbol{\sigma}_j^{\preccurlyeq})\!\downarrow$. For $\boldsymbol{\tau} \in A_j \cap \boldsymbol{\sigma}_j^{\preccurlyeq}$ we let $Q_{\boldsymbol{\tau}}$ be the set of $\boldsymbol{\zeta} \in D_j^{\preccurlyeq} \cap T^p\!\downarrow$ such that either:

- $\boldsymbol{\tau}\!\downarrow \not\preccurlyeq \boldsymbol{\zeta}$; or
- in $T^p(\boldsymbol{\zeta})$ there are G_0 and G_1, $3^j g$-big above τ_n, which $\Gamma(\boldsymbol{\zeta}, -)$-split mod $B^p(\boldsymbol{\zeta})$.

Then Lemma 3.4.19 says that for all $\boldsymbol{\mu} \in D_j^{\preccurlyeq} \cap T^p\!\downarrow$ the set $Q_{\boldsymbol{\tau}}$ is $3^j \boldsymbol{g}$-big above $\boldsymbol{\mu}$. By Lemma 3.4.11, $Q_{\boldsymbol{\tau}}$ is $3^j \boldsymbol{g}$-big above $D_j^{\preccurlyeq} \cap T^p\!\downarrow$. By Lemma 3.4.10, $\bigcap_{\boldsymbol{\tau} \in A_j \cap \boldsymbol{\sigma}_j^{\preccurlyeq}} Q_{\boldsymbol{\tau}}$ is $3^j \boldsymbol{g}$-big above D_j. Thus, we can find $E_{j,0}$ and $E_{j,1}$, finite subsets of T^p which are uniformly $3^j \boldsymbol{g}$-big above $A_j \cap \boldsymbol{\sigma}_j^{\preccurlyeq}$, which locally Γ-split mod B^p. We obtain F_j as before. Applying Lemma 3.4.20 we finally get $F'_j \subset A_j^{\preccurlyeq} \cap (\boldsymbol{\sigma}^*)^{\preccurlyeq}$, 3^{j-1}-big above $\boldsymbol{\sigma}^*$, and $E'_j \subset A_j^{\preccurlyeq} \cap \boldsymbol{\sigma}_j^{\preccurlyeq}$, $3^{j-1}\boldsymbol{g}$-big above $\boldsymbol{\sigma}_j$, which Γ-split mod B^p.

In this proof we employ the following notation: for a set $X \subset (\omega^{<\omega})^n$ and $k \leqslant n$ we let $X_k = \mathrm{dom}_k X$. To define a set X it suffices to first define X_1; then, for all $\tau_1 \in X_1$, define $X_2(\tau_1)$ (a set of strings); then, for all $(\tau_1, \tau_2) \in X_2$, define $X_3(\tau_1, \tau_2)$, and so on.

We define the set A_{j-1}. First, we consider all $\boldsymbol{\sigma} \in C^*$ such that $\sigma_1 \neq \sigma_1^*, (\sigma_j)_1$. For all such $\boldsymbol{\sigma}$ we let $A_{j-1} \cap \boldsymbol{\sigma}^{\preccurlyeq} = A_j \cap \boldsymbol{\sigma}^{\preccurlyeq}$. We let $A_{j-1,1} \cap (\sigma_1^*)^{\preccurlyeq} = (F'_j)_1$ and $A_{j-1,1} \cap ((\sigma_j)_1)^{\preccurlyeq} = (E'_j)_1$.

Next, consider all $\boldsymbol{\sigma} \in C^*$ such that $\sigma_1 = \sigma_1^*$, but $\sigma_2 \neq \sigma_2^*$. For all $\tau_1 \in (F'_j)_1$ we let $A_{j-1}(\tau_1) \cap (\boldsymbol{\sigma}\!\upharpoonright_{(1,n]}) = A_j(\tau_1^{-A_{j,1}}) \cap (\boldsymbol{\sigma}\!\upharpoonright_{(1,n]})$; this completely defines $A_{j-1} \cap \boldsymbol{\sigma}^{\preccurlyeq}$. We similarly define $A_{j-1} \cap \boldsymbol{\sigma}^{\preccurlyeq}$ for

$\boldsymbol{\sigma} \in C^*$ such that $\sigma_1 = (\sigma_j)_1$ but $\sigma_2 \neq (\sigma_j)_2$. Then, for all $\tau_1 \in (F'_j)_1$ we let $A_{j-1,2}(\tau_1) = (F'_j)(\tau_1)$; this defines $A_{j-1,2} \cap (\boldsymbol{\sigma}^* \restriction_2)^{\preccurlyeq}$, and similarly define $A_{j-1,2} \cap (\boldsymbol{\sigma}_j \restriction_2)^{\preccurlyeq}$. The process continues similarly until all of A_{j-1} is defined.

The case $k > 0$ is very similar. Morally it follows the idea of the proof of Lemma 3.3.27, extending bushily on the first k coordinates so that we can emulate the proof of the case $k = 0$ (but with $n - k$ replacing n) within the image. We give a sketch. Again we work by induction on $|C|$; we start with some C for which we inductively already have A as required; and add to C a tuple $\boldsymbol{\sigma}^*$ to get C^*. We now let the list $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2, \ldots, \boldsymbol{\sigma}_m$ contain those elements $\boldsymbol{\sigma} \in C$ such that $\boldsymbol{\sigma} \restriction_k = \boldsymbol{\sigma}^* \restriction_k$ but $\sigma_{k+1} \neq \sigma^*_{k+1}$. We start with $A_m = A \cup \{\boldsymbol{\sigma}^*\}$ and build sets A_j with the same properties as above. Given A_j we aim to find $E_{j,0}, E_{j,1}$ and F_j as above, except that we also require that $\operatorname{dom}_k E_j = \operatorname{dom}_k F_j$; this is possible because $\boldsymbol{\sigma}_j \restriction_k = \boldsymbol{\sigma}^* \restriction_k$: we first get E_j as above, and then extend $\operatorname{dom}_k E_j$ to $\operatorname{dom}_k F_j$; and "relabel" E_j by letting $E_j(\zeta) = E_j(\tau)$ for all $\zeta \in \operatorname{dom} F_j$ extending $\tau \in \operatorname{dom} E_j$. Then we obtain E'_j and F'_j but require that $\operatorname{dom}_k E'_j = \operatorname{dom}_k F'_j = \operatorname{dom} F_j$; we apply Lemma 3.4.20 within $T^p(\boldsymbol{\zeta})$ for each $\boldsymbol{\zeta} \in \operatorname{dom} F_j$. We then define A_{j-1} as above. □

Proposition 3.4.22. *Every condition in* $\mathbb{P}_n$ *forces that* $\deg_{\mathrm{T}}(\boldsymbol{x}^G)$ *is a strong minimal cover of* $\deg_{\mathrm{T}}(\boldsymbol{x}^G{\downarrow})$.

Proof. Let $p \in \mathbb{P}_n$ which strongly forces that $\Gamma(\boldsymbol{x}^G)$ is total, and forces that it is not computable from $\boldsymbol{x}^G{\downarrow}$. Fix $k \in \{0, 1, \ldots, n-1\}$.

Using Lemma 3.4.21 and the by now familiar construction we obtain an extension q of p which (strongly) forces that $\Gamma(\boldsymbol{x}^G) \oplus (\boldsymbol{x}^G \restriction_k) \geqslant_{\mathrm{T}} x^G_{k+1}$. Iterating for each k we obtain a condition which forces that $\Gamma(\boldsymbol{x}^G) \equiv_{\mathrm{T}} \boldsymbol{x}^G$. □

3.5 Proof of the main theorem

We prove Theorem 3.1.1. We have obtained a directed sequence of forcing notions

$$\begin{tikzcd} \mathbb{Q}_1 & \mathbb{Q}_2 \arrow[l, "i_2"'] & \mathbb{Q}_3 \arrow[l, "i_3"'] & \mathbb{Q}_4 \arrow[l, "i_4"'] & \cdots \arrow[l, "i_5"'] \end{tikzcd}$$

With each i_n a restrction map. For $m < n$ let $i_{n\to m} = i_{m+1} \circ i_{m+2} \circ \cdots \circ i_n$ (and of course let $i_{n\to n} = \mathrm{id}_{\mathbb{Q}_n}$). A composition of restriction maps is a restriction map, so each $i_{n\to m}$ is a restriction map.

As sets, the forcing notions $\mathbb{Q}_n$ are pairwise disjoint. Let $\mathbb{Q}_{<\omega} = \bigcup_n \mathbb{Q}_n$. We order $\mathbb{Q}_{<\omega}$ as follows: if $p \in \mathbb{Q}_n$ and $q \in \mathbb{Q}_m$ then q extends p if $m \geqslant n$ and $i_{m\to n}(q) \leqslant p$ in $\mathbb{Q}_n$. Note that the ordering on each $\mathbb{Q}_n$ agrees with this ordering.

For $n < \omega$ let $\mathbb{Q}_{\leqslant n} = \bigcup_{m\leqslant n} \mathbb{Q}_m$, ordered as a sub-order of $\mathbb{Q}_{<\omega}$. Define $j_{\omega\to n} \colon \mathbb{Q}_{<\omega} \to \mathbb{Q}_{\leqslant n}$ by letting, for $q \in \mathbb{Q}_m$, $j_{\omega\to n}(q) = q$ if $m \leqslant n$, and otherwise $j_{\omega\to n}(q) = i_{m\to n}(q)$. For $m \geqslant n$ let $j_{m\to n} \colon \mathbb{Q}_{\leqslant m} \to \mathbb{Q}_{\leqslant n}$ be $j_{\omega\to n} \restriction_{\mathbb{Q}_{\leqslant m}}$. These maps are restriction maps and they commute: for $n \leqslant m \leqslant \alpha \leqslant \omega$, $j_{\alpha\to n} = j_{m\to n} \circ j_{\alpha\to m}$.

Let $G_{<\omega} \subset \mathbb{Q}_{<\omega}$ be very generic. Let $G_{\leqslant n}$ be the filter in $\mathbb{Q}_{\leqslant n}$ generated by the generic directed set $j_{\omega\to n}[G_{<\omega}]$. By Lemma 3.4.15, each $\mathbb{Q}_n$ is dense in $\mathbb{Q}_{\leqslant n}$; so $G_n = G_{\leqslant n} \cap \mathbb{Q}_n$ is a fairly generic filter of $\mathbb{Q}_n$; and $i_{m\to n}[G_m] \subseteq G_n$. (By 'very generic' and 'fairly generic' we mean that if we need G_n to be sufficiently generic, then we can ensure that by making $G_{<\omega}$ sufficiently generic. Technically, for any countable collection $\mathcal{D}$ of dense subsets of $\mathbb{Q}_n$ we can find a countable collection $\mathcal{E}$ of dense subsets of $\mathbb{Q}_{<\omega}$, such that if $G_{<\omega}$ meets every set in $\mathcal{E}$, then G_n meets every set in $\mathcal{D}$.)

This gives us a sequence $x_1, x_2, \ldots$ of elements of Baire space such that $(x_1, \ldots, x_n) = \boldsymbol{x}^{G_n}$. By Proposition 3.4.22, each tuple $(x_1, \ldots, x_n)$ is a strong minimal cover of $(x_1, \ldots, x_{n-1})$; and $x_n \in \mathrm{DNC}^{(x_1,\ldots,x_{n-1})}$.

Chapter 4

Multiple genericity

4.1 Introduction

Real numbers that are typical in some sense play an important role in computability theory. They arise from considering the notions of measure and category. Given a countable collection of sets that are considered large in some sense, we consider the reals that are in every such set in some way. If the sets in our collection are large with respect to measure, we arrive at *random* reals, and if they are large with respect to category, we arrive at *generic* reals.

In [26], Jockusch introduced the notion of n-generic reals, for every $n \in \omega$. The original definition was in terms of reals that are generic for Cohen forcing restricted to n-quantifier arithmetic. This was shown by Jockusch and Posner ([26]) to be equivalent to the following. Our collection of sets are the Σ^0_n sets of strings. We say that the real A *meets* the Σ^0_n set S if there is some $\sigma \prec A$ such that $\sigma \in S$, and that A *avoids* S if there is some $\sigma \prec A$ such that no extension of σ is in S. Then a real is n-generic if it either meets or avoids every Σ^0_n set of strings. The n-generic sets form a proper hierarchy: every $n+1$-generic real is also n-generic, but the converse does not hold.

We also have the *weakly* n-generic sets, as defined by Kurtz in [34]. We say that a set of strings S is *dense* if every string has an extension in S. Then a real A is weakly n-generic if it meets every dense Σ^0_n set of strings. Kurtz ([34]) showed that this refines the hierarchy, in that we have for all n, the n-generic reals properly contain the weakly $n+1$-generic reals, which properly contain the $n+1$-generic reals.

The greater the n, the more typical we consider an n-generic real to be. In many cases, typical behaviour starts with the 2-generic reals, and will fail for 1-generic reals. A similar situation occurs with the hierarchy of n-random sets. Barmpalias, Day, and Lewis ([2]) survey many such results for randomness and genericity. As an example, Kurtz ([34]) showed that the collection of sets computing a 2-generic real has measure 0, whereas the collection of sets computing a 1-generic real has measure 1 ([34]). Thus it is of great interest to determine exactly when typical behaviour starts, and we may use notions of genericity intermediate between 1- and 2-genericity to more finely specify this.

Several such notions have already been defined. Apart from weak 2-genericity, the most well-known notion is *pb-genericity*, which was introduced by Downey, Jockusch, and Stob in [20]. We meet *pb-dense* sets of strings. A set is pb-dense if it is the range of a total function $f : 2^{<\omega} \to 2^{<\omega}$ with $f(\sigma) \succcurlyeq \sigma$ which can be approximated, as in the Limit Lemma, with a primitive recursive bound on the number of mind changes. Schaeffer ([40]) defines *dynamic genericity*, which is a stronger notion than pb-genericity which also uses dense sets of strings.

To highlight the difference between these notions, we consider what we must do in order to construct these generics. Suppose that we would like to construct a real A which is 1-generic. Let S be a c.e. set of strings. Suppose that we already have decided that σ will be an initial segment of our real A. For A to meet or avoid S, we simply wait until a string τ with $\tau \succ \sigma$ is enumerated into S. If no such τ exists, then σ avoids S. If such a τ does exist, we can then let A extend τ, and A will meet S. The point is that we need to act at most once in order to satisfy the requirement that we meet or avoid S.

Suppose we would like to instead make A pb-generic. Let $f : 2^{<\omega} \to 2^{<\omega}$ be a total function with approximation $\langle f_s \rangle$ that has a primitive recursive bound p on the number of mind changes. Suppose that we decide at stage s that our real A must extend σ. So that A meets range f, at every stage t of the construction after stage s, we would like A_t, our current approximation to A, to extend $f_t(\sigma)$, the current approximation to $f(\sigma)$. Although we do not know during the construction which approximation $f_s(\sigma)$ to $f(\sigma)$ is correct, we do know that we only have to change our approximation to A at most $p(\sigma)$ many times before it will permanently extend $f(\sigma)$. In particular, given the string σ and the function f with bound p, the number of times we must act in order to satisfy the requirement that some extension of σ meets range f is bounded in advance, namely by $p(\sigma)$.

We wish to generalise this notion of genericity so that given σ and some set of strings T we would like to meet or avoid, the number of times we must act in order to meet T is only revealed to us during the course of the construction. We would also like to allow for the possibility that T is not a dense set of strings.

We imagine, as is standard in computability theory, that we are trying to construct a real that is generic in this sense, and that our opponent gets to play the set of strings T. Suppose that we have decided that σ is an initial segment of the real we are building. Our opponent is enumerating the strings in T. We look for such a string extending σ. Suppose that at some stage s our opponent enumerates the string τ extending σ. We then let our approximation to A extend τ. For the moment we think that we have met T, and will work towards meeting other such sets U played by the opponent. At some later stage, our opponent then enumerates some extension τ' of τ, and demands that we must instead ensure that our real A extends τ' in order to meet T. We change our approximation to A to extend τ'. This can repeat a certain number of times. We require however, that our opponent tells us *at stage s* how many times this may repeat.

This is formalised as follows. The set T is thought of as the range of a partial computable function $f : \omega \times \omega \to 2^{<\omega}$, equipped with a partial computable function h. Suppose at stage s we see $f(x,0)[s]\downarrow$. We also require that $h(x)[s]\downarrow$. Then $f(x,0)$ is the current string which we would like to extend in order to meet T, and we know that we may see up to $h(x)$ many extensions of $f(x,0)$. We may at some later stage s' see that $f(x,1)[s']\downarrow$, in which case we would then like to extend $f(x,1)$ in order to meet T. Our opponent must ensure that $|\{k : f(x,k)\downarrow\}| < h(x)$. Of course there may also be $y \neq x$ such that $f(y,0)[s']\downarrow$. In order for the real A to meet T, we only need there to be some x such that $f(x,k) \prec A$, where k is greatest such that $f(x,k)\downarrow$. We say that the real A avoids range f if it avoids range f when considered simply as a c.e. set of strings. That is, if there is some $\sigma \prec A$ such that there are no x and k such that $f(x,k) \not\succ \sigma$. We refer to range f as an *ω-change set of strings*. We give the definition again in the next section.

In section 4.3 we consider the computational power required to compute such a generic. We see a connection between these generics and the hierarchy recently introduced by Downey and Greenberg in [17]. In section 4.4 we first present a corrected version of the proof from [13] that if a set has no c.e. tight cover then it is computable in a 1-generic, and then extend this result to the case of ω-change generics. In the final section, we give another proof of Haught's theorem that every noncomputable degree below a 1-generic below $\varnothing'$ contains a 1-generic. We hope in further work to show that the analogous result for ω-change generics does not hold, and that they are not downward dense below $\varnothing'$.

4.2 Definitions

An *ω-change set of strings* is the range of a partial computable function $f : \omega \times \omega \to 2^{<\omega}$ for which there is a partial computable function $h : \omega \to \omega$ such that for all x,

1. if $f(x, k+1)\downarrow$ then $f(x,k)\downarrow$,
2. if $f(x, k+1)\downarrow$ then $f(x, k+1) \succcurlyeq f(x,k)$,
3. for all s, if $f(x,0)[s]\downarrow$, then $h(x)[s]\downarrow$, and
4. $|\{k : f(x,k)\downarrow\}| < h(x)$.

Let range f be an ω-change set of strings with partial computable bound h. We say that a set A *meets* range f if there is some x and k such that $f(x,k) \prec A$ and $f(x,k')\uparrow$ for all $k' > k$. We say that A *avoids* range f if A avoids range f as a c.e. set of strings. That is, there is some $\sigma \prec A$ such that for all x and n, $f(x,n) \not\succ \sigma$. We say that a set is ω-change generic if it meets or avoids every ω-change set of strings.

We see that every ω-change generic is 1-generic, as every c.e. set of strings is an ω-change set of strings. If range f is an ω-change set of strings, then range f is a Σ^0_2 set of strings. Therefore every 2-generic is ω-change generic.

For the remainder of this chapter, we let $\langle f_i, h_i \rangle_{i<\omega}$ be an effective list such that $\langle \text{range } f_i \rangle$ is a list of all ω-change sets of strings, and that range f_i has partial computable bound h_i.

4.3 Computing ω-change generics

In this section we give several results which progressively refine the computational power required to compute ω-change generic sets.

4.3.1 Forcing arguments

Importantly, the following theorem shows that ω-change generics exist below $\varnothing'$. Note that this is not the case for weakly 2-generic sets.

Theorem 4.3.1. *There is an ω-change generic set G with $G \leqslant_T \varnothing'$.*

Proof. We must built $G \leqslant_T \varnothing'$ and satisfy the requirements

R_e**:** G meets or avoids range f_e.

We build G by finite extension. The construction will proceed in stages, with each stage consisting of possibly many substages. Let G_s be the string at the end of the last substage of stage s, and let $G_{s,n}$ denote the string at the n^{th} substage of stage s. Let $G_0 = \lambda$, the empty string.

Construction

Stage $e + 1$: we deal with R_e at this stage.

Substage 0: Let $G_{e+1,0} = G_e$.

Substage 1: We ask $\varnothing'$ the Σ^0_1 question

$$(\exists s)(\exists x)(\exists k)(h_e(x)[s]\downarrow \text{ and } f_e(x,k)[s]\downarrow \succ G_{e+1,0}).$$

If the answer is no, then there is no string in range f_e which extends $G_{e+1,0}$, so we can let $G_{e+1} = G_{e+1,0}$ and proceed to the next stage. If the answer is yes, then let x_e be the least x found at the least such stage. We let $G_{e+1,1} = f_e(x_e,k)$ where k is greatest such that $f_e(x_e,k)[s]\downarrow$, and proceed to the next substage.

Substage u, $u \geqslant 2$: We ask $\varnothing'$ the Σ^0_1 question

$$(\exists k)(\exists s)(f_e(x_e,k)[s]\downarrow\succ G_{e+1,u-1}).$$

If the answer is no we let $G_{e+1} = G_{e+1,u-1}$ and proceed to the next stage. If the answer is yes, we let $G_{e+1,u} = f_e(x_e,k)$, and proceed to the next substage.

There are at most $h_e(x_e)$ substages of stage e as

$$|\{k : f_e(x,k)\downarrow\}| < h_e(x).$$

In this way we make sure that $G_e = f_e(x_e,k)$ where k is the greatest such that $f_e(x_e,k)\downarrow$, and so G meets range f_e, and R_e is satisfied. □

Our next step is showing that every $\overline{\mathrm{GL}_2}$ degree computes an ω-change generic. Recall that a set A is in $\overline{\mathrm{GL}_2}$ if it is not generalised low$_2$. That is, for each function $f \leqslant_T A \oplus \varnothing'$ there is a function

$g \leqslant_T A$ such that $g(n) \geqslant f(n)$ for infinitely many n; see Corollary 2.23.8 of [18]. The proof is very similar to that of Lemma 2.24.23 of [18], which itself is fairly similar to Lemma 3 in [28].

Theorem 4.3.2. *Every $\overline{GL_2}$ degree computes an ω-change generic.*

Proof. Let $\boldsymbol{a}$ be a $\overline{\mathrm{GL}_2}$ degree. We build an ω-change generic $G \leqslant_T \boldsymbol{a}$ and satisfy the requirements

R_e**:** G meets or avoids range f_e.

First let us define the function $p : \omega \times 2^{<\omega} \times \omega \to \omega$. We let $p(e, \sigma, s) = 0$ if there is no x and k such that $f_e(x, k)[s] \downarrow \succ \sigma$. Otherwise, we let $s_{e,\sigma} \leqslant s$ be the least such that there are x and k such that $f_e(x, k)[s_{e,\sigma}] \downarrow \succ \sigma$. Let $x_{e,\sigma}$ be the least such x at stage $s_{e,\sigma}$. Then if k is greatest such that $f_e(x_{e,\sigma}, k)[s] \downarrow$, we let $p(e, \sigma, s)$ be the least stage $s' \leqslant s$ such that $f_e(x_{e,\sigma}, k)[s'] \downarrow$. We let $q(e, \sigma) = \lim_s p(e, \sigma, s)$. As can be seen from the proof of the previous theorem, $q \leqslant_T \varnothing'$. Let

$$l(n) = \max \{ q(e, \sigma) : e, |\sigma| \leqslant n \}.$$

We also have $l \leqslant_T \varnothing'$. Let $A \in \overline{\mathrm{GL}_2}$. As $\varnothing' \leqslant_T A \oplus \varnothing'$, $l \leqslant_T A \oplus \varnothing'$. By Corollary 2.23.8 in [18], there exists a function $g \leqslant_T A$ that escapes domination by l. That is, $(\exists^\infty s)(g(s) > l(s))$. We assume without loss of generality that g is nondecreasing and unbounded.

We obtain G as $\cup_s G_s$, where each G_s is a string of length s. At stage s we look for extensions to our current approximation G_{s-1} in range f_e for $e \leqslant s$. The bound on the search is given by $g(s)$. At stage s, if R_e is the strongest priority requirement that is not currently satisfied, we see whether there is $f_e(x,k)[g(s)] \downarrow \succ G_{s-1}$. If so, we choose x to be the least such at stage s, and k the greatest such for this x. We will then start working towards $f_e(x,k)$ one bit at a time. Once we reach a stage t where $G_t = f_e(x,k)$, we say that R_e is satisfied at stage s. This attack can be interrupted by any requirement of stronger priority. If at any stage $u > s$ we see that there is $k' > k$ such that $f_e(x,k')[g(u)] \downarrow$, we say that R_e is unsatisfied at stage u. We then repeat the strategy for S_e.

The construction is carried out computably in g, and so $G \leqslant_T A$. It is clear that if R_e is permanently satisfied then G meets range f_e. We show that each R_e is satisfied. Suppose by induction that all R_i for $i < e$ are permanently satisfied at all stages after stage s_0. Let $s > s_0$ be such that $g(s) > l(s)$. If an attack as above is started at stage s, then this will lead to R_e being permanenty satisfied. Otherwise, there must be a stage $v \in [s_0, s)$ such that an attack for R_e is started at stage v. Suppose that this attack works towards the string $f_e(x,k)$. Then as R_e is not said to be unsatisfied at any stage before s, we must have for all $t \in (g(v), g(s)]$ that the greatest l such that $f_e(x,l)[t] \downarrow$ is k. Then $g(v) \geqslant q(e, G_{v-1})$ and so R_e will be satisfied at all stages after stage s.

□

Downey, Jockusch, and Stob in [20] extended the result of Jockusch and Posner from [28] that every $\overline{\mathrm{GL}_2}$ degree bounds a 1-generic to show that every array noncomputable degree bounds a 1-generic.

This is significant because array noncomputable degrees can exist within the generalized low$_2$, and in fact the low, degrees ([19], [20]).

We can too show that ω-change generics exist in the generalized low$_2$ degrees. We require some concepts introduced by Downey and Greenberg in [17]. They define for all ordinals $\alpha \leqslant \varepsilon_0$ the notion of an α-c.a. function. The concepts are quite sensitive to the ordinal notations used. We refer the reader to Chapter II of [17] for a discussion of this. The ordinals must have an effective Cantor normal form. In this chapter we deal only with the ordinals ω^2 and ω, and so for our purposes, if $\beta < \omega^2$, then $\beta = \omega k + n$, and we *know* k and n. We consider ω^2 with the lexicographic ordering, and so if $\alpha, \beta < \omega^2$, then $\alpha = \omega l + m$ and $\beta = \omega k + n$ for some l, m, k, n, and $\alpha < \beta$ if and only if $l < k$, or $l = k$ and $m < n$.

Definition 4.3.3. An ω^2-computable approximation of a function f is a computable approximation $\langle f_s \rangle$ of f, equipped with a uniformly computable sequence $\langle o_s \rangle_{s<\omega}$ of functions from ω to ω^2 such that for all x and s,

- $o_{s+1}(x) \leqslant o_s(x)$, and
- if $f_{s+1}(x) \neq f_s(x)$, then $o_{s+1}(x) < o_s(x)$.

Then a function $f : \omega \to \omega$ is ω^2-c.a. if it has an ω^2-computable approximation. We say that a degree $\boldsymbol{a}$ is ω^2-c.a. dominated if every function computable in $\boldsymbol{a}$ is dominated by some ω^2-c.a. function. There is a uniform version of this notion as well. We say that a degree $\boldsymbol{a}$ is uniformly ω^2-c.a. dominated if there is some ω^2-c.a. function which dominates every function computable in $\boldsymbol{a}$.

The definition of an ω-c.a. function is similar, replacing ω^2 with ω, and considering ω with its standard order. A function is ω-c.a. if and only if it is weak truth-table below $\emptyset'$.

Theorem 4.3.4. *If $\boldsymbol{a}$ is not uniformly ω^2-c.a. dominated then A bounds an ω-change generic set.*

Proof. We show that the function l from Theorem 4.3.2 is ω^2-c.a. We first show that the function q from Theorem 4.3.2 is $(\omega+1)$-c.a. We have $q(e,\sigma) = \lim_s p(e,\sigma,s)$ and p is a total computable function. We define a sequence of functions $\langle o_s \rangle_{s<\omega}$, $o_s : \omega \times 2^{<\omega} \to \omega + 1$. Let $o_s(e,\sigma) = \omega$ if $p(e,\sigma,s) = 0$. If $p(e,\sigma,s) \neq 0$, then $s_{e,\sigma}$ and $x_{e,\sigma}$ are defined at stage s, and $h_e(x_{e,\sigma})[s]\downarrow$. We let $o_s(e,\sigma) = h_e(x_{e,\sigma}) + 1 - n$, where n is the number of different numbers in the list $p(e,\sigma,s_{e,\sigma}), p(e,\sigma,s_{e,\sigma}+1), \ldots, p(e,\sigma,s)$.

The function l has the computable approximation $\langle l_s \rangle$ where

$$l_s(n) = \max\{\, p(e,\sigma,s) : e, |\sigma| \leqslant n\}.$$

There are $(n+1)(2^{n+1}-1)$ many pairs (e,σ) such that $e \leqslant n$ and $|\sigma| \leqslant n$. Let $o'_s(n) = (\ (n+1)(2^{n+1}-1) - k\ ,\ \sum_{e\leqslant n, |\sigma|\leqslant n} o_s(e,\sigma)\ )$ where k is the number of pairs (e,σ) such that $p(e,\sigma,s) \neq 0$ and we do not include terms in the summation where $o_s(e,\sigma) = \omega$. For any n, $o'_s(n)$ is nonincreasing, as the number of pairs (e,σ) with $p(e,\sigma,s) \neq 0$ can only increase, and $o_s(e,\sigma)$ is nonincreasing. If for some s we have $l_{s+1}(n) \neq l_s(n)$, this must be because there is some pair (e, σ) such that either $p(e, \sigma, s+1) > p(e, \sigma, s)$, in which case $o_s(e, \sigma)$ will decrease and so will $o'_s(n)$, or $p(e, \sigma, s) = 0$ and $p(e, \sigma, s+1) \neq 0$, in which case the first coordinate of $o'_s(n)$ will decrease, and so $o'_s(n)$ will.

Let $\boldsymbol{a}$ be a not uniformly ω^2-c.a. dominated degree. Then $\boldsymbol{a}$ computes a function which escapes domination by l. Then as in the proof of Theorem 4.3.2, we can build a ω-change generic set below $\boldsymbol{a}$. $\square$

Corollary 4.3.5. *There exist ω-change generic sets in the generalized low$_2$ degrees.*

Proof. Downey and Greenberg's hierarchy of α-c.a. dominated degrees is contained within the generalized low$_2$ degrees. They show that if α is a power of ω, then there is a degree which is α-c.a. dominated but not uniformly α-c.a. dominated. See Chapter III, and in particular Section III.5 of [17] for more details.

Taking a degree which is ω^2-c.a. dominated but not uniformly ω^2-c.a. dominated, by the previous theorem, there is an ω-change generic set in the generalized low$_2$ degrees. $\square$

4.3.2 A c.e. permitting argument

A closely related notion from [17] is that of a totally α-c.a. degree. We say that the degree $\boldsymbol{a}$ is totally α-c.a. if every function computable in $\boldsymbol{a}$ is α-c.a.

For c.e. degrees, the notions of (uniformly) α-c.a. dominated and (uniformly) totally α-c.a. coincide (see Section III.5 of [17]).

In Downey and Greenberg's hierarchy, immediately below the uniformly totally ω^2-c.a. degrees are the totally ω-c.a. degrees. Therefore we can slightly improve the previous result for c.e. degrees.

Theorem 4.3.6. *Every not totally ω-c.a. c.e. degree computes an ω-change generic set.*

Proof. Let $\boldsymbol{a}$ be a not totally ω-c.a. c.e. degree, $A \in \boldsymbol{a}$ be a c.e. set, and let $g = \Gamma(A)$ be a function that is not ω-c.a. (where Γ is a Turing reduction). Let $\gamma_s(n)$ be the use at stage s of computing $g(n)$ via $\Gamma(A)$. We have a Δ^0_2-approximation g_s for g that is generated from an enumeration of A via Γ.

We build a set G and a Turing functional Δ such that $G = \Delta(A)$ to satisfy the requirements

R_e**:** G either meets or avoids range f_e.

We first consider how to satisfy R_e in the simplified case where for all x, if $h_e(x)\downarrow$ then $h_e(x) = 1$. This situation is equivalent to meeting or avoiding a c.e. set of strings. The construction proceeds in stages $s \in \omega$. Associated with every requirement at every stage is a finite sequence of natural numbers we call *lengths*. If l_i is a length, then the i^{th} substrategy for R_e will seek at stage s an extension to $G_{s-1} \upharpoonright l_i$ in range $f_{e,s}$. If we find an appropriate extension to $G_{s-1} \upharpoonright l_i$ (we say that l_i is *realized*) we would like to change our approximation to G to meet the extension, but will require permission from A to do so. Whenever a new length is defined we assign to it a *permitting number*; in this instance we assign l_i the permitting number i. As l_i has permitting number i, we will grant permission for l_i if we see a change in the Δ^0_2-approximation to $g(i)$, and hence a change in A below $\gamma_s(i)$. If permission is granted, we change our approximation to G to meet the extension, and the requirement will be permanently satisfied. As we may never receive permission on any of the lengths already in our sequence, we choose a fresh large number to be a new length and assign it a permitting number. If there are infinitely many realized lengths, none of which receive permission, then we derive a contradiction to g not being ω-c.a. as follows. Suppose l_i is realized at stage s_i. As we do not receive permission for l_i after stage s_i, we know that $g(i)$ cannot change past this stage. Thus we can computably bound the number of times $g(i)$ can change, which is a contradiction. In fact, in this simplified case we have that $A \upharpoonright \gamma_{s_i}(i) = A_{s_i} \upharpoonright \gamma_{s_i}(i)$ for all i, which contradicts A being noncomputable.

Unfortunately, we do not have $h_e(x) = 1$ for all x such that $h_e(x)\downarrow$. This will mean that a single length may require multiple permissions, as we now discuss. We begin as above, with the length l_0 with permitting number 0. Suppose at stage s, l_0 is realized when we see for some x that $f_e(x,0)[s]\downarrow$, $h_e(x)[s]\downarrow$, and $G_{s-1} \upharpoonright l_0 \prec f_e(x,0)$. We refer to the x here as the *location* for this attack on R_e, and associate x with l_0 by letting $x_0 = x$. As above, while we wait to receive permission for l_0 we will choose a new length l_1 with permitting number 1. If we were to ever receive permission for l_0 at stage t, then we would change G_t to extend $f_e(x_0, 0)$. In the simplified case, this action would have been enough to permanently satisfy R_e. Here in the ω-change generic case, at some later stage t' we may have $f_e(x_0, 1)[t'] \downarrow \succ f_e(x_0, 0)$ but $f_e(x_0, 1) \not\prec G_{t'}$, in which case R_e would no longer appear to be satisfied at stage t'. To change our approximation to G again we would require another permission for l_0. The number of permissions we may require for l_0 is at most $h_e(x_0)$, since $|\{k : f_e(x_0, k) \downarrow\}| < h_e(x_0)$. If not enough permissions for l_0 are received to satisfy R_e, then we can computably bound the number of times $g(0)$ may change as follows. We can approximate $g(0)$ every time l_0 receives permission, and as this occurs strictly fewer than $h_e(x_0)$ many times, our approximation changes at most $h_e(x_0)$ many times.

There is one last complication. Suppose, as above, we acted for l_0 to have G_t meet $f_e(x_0, 0)$. R_e would then seem to be satisfied. We may have no further need for l_1, and so we clear it from our sequence of lengths. The problem is that we must challenge the non-ω-c.a.-ness of g at *all* $i \in \omega$. What should we do with changes in $g(1)$? The solution is to let l_0 receive permission from A when there is a change in $g(0)$ *or* $g(1)$. If at some later stage t' we see that $f_e(x_0, 1)[t'] \downarrow \succ f_e(x_0, 0)$, we will wait for permission for l_0, choose a new length l_1, and assign it permitting number 2. Why could we not use the old length l_1 here? We would like to be able to reason that if we find no extension to $G_{t'} \upharpoonright l_1$ then $G_{t'}$ avoids range f_e. However, if we were to use the old value for l_1 we might have $l_1 < |f_e(x_0, 1)|$.

Therefore we choose a fresh number for l_1.

In general, each length l_i will be assigned a *permitting number* n. If l_i has permitting number n, it receives permission when there is a change in the approximation to $g(n)$. If l_i receives permission we clear all $l_{i'}$ for $i' > i$ and then l_i takes responsibility for tracking the changes in g on the permitting numbers of the $l_{i'}$ – the permitting number for l_i will then become a permitting *interval*. This is achieved in practice by lifting the use $\delta_{t+1}(l_i)$ to equal $\gamma_{t+1}(n_{max})$, where n_{max} is the largest current permitting number. We show in Lemma 4.3.7 that the use can be lifted only $h_e(x_i)$ many times, and so is well-defined. To argue that R_e is met, we show that if there are infinitely many realized lengths, none of which receive enough permissions, then we will be able to obtain a contradiction by building an ω-c.a. approximation $\hat{g}$ for g. The definition of $\hat{g}$, as well as a bound on the number of times the approximation may change, will be given during the verification. We now give formal details of the construction.

At every stage s and for every requirement R_e we will have a sequence

$$l_{e,0,s}, l_{e,1,s}, \dots, l_{e,i_{max}(e,s),s}$$

of *lengths*, and a sequence

$$x_{e,0,s}, x_{e,1,s}, \dots, x_{e,i_{max}(e,s)-1,s}$$

of *locations*. At some stages we will also define $x_{e,i_{max}(e,s),s}$. During the construction we may clear all or parts of the sequences, and so if at stage s we have i entries in our sequence of lengths for R_e, we let $i_{max}(e,s) = i-1$; when the sequence is empty we let $i_{max}(e,s)$ be undefined. We will say that the length $l_{e,i,s}$ is *waiting* if $x_{e,i,s}$ is defined and we are currently seeking permission to change our approximation to G to extend $f_e(x_{e,i,s}, k)$, where k is greatest such that $f_e(x_{e,i,s}, k)[s]\downarrow$.

To every length $l_{e,i,s}$ we associate a *permitting interval $I(l_{e,i,s})$*. This is an interval of natural numbers; its left end is fixed from when $l_{e,i,s}$ is first defined, but its right end may grow with time (but only finitely often). We say that $l_{e,i,s}$ *is permitted* at stage s if for some $n \in I(l_{e,i,s})$ we have $g_{s-1}(n) \neq g_s(n)$.

We say that R_e is satisfied stage s if $x_{e,i_{max}(e,s),s}$ is defined, and $f_e(x_{e,i_{max}(e,s),s}, k) \prec G_s$, where k is greatest such that $f_e(x_{e,i_{max}(e,s),s}, k)[s]\downarrow$.

We say that R_e requires attention at stage s if one of the following holds:

1. the sequence of lengths for R_e is empty at stage s.
2. For some i such that $l_{e,i,s}$ is waiting, $l_{e,i,s}$ is permitted at stage s.
3. R_e was satisfied at some previous stage, but is not satisfied at stage s.
4. R_e is not satisfied at stage s, and there is an x not occuring in our sequence of locations at stage s, and a k such that $h_e(x)[s]\downarrow$ and $f_e(x,k)[s]\downarrow \succ G_s \upharpoonright l_{e,i_{\max}(e,s),s}$.

Why we may want R_e to receive attention in case (2), even when

R_e is satisfied, bears some explanation. Suppose at stage s we have that R_e is satisfied but that there is $i < i_{max}(e, s)$ such that $l_{e,i,s}$ is waiting and $l_{e,i,s}$ is permitted. Suppose that $n \in I(l_{e,i,s})$. We would then proceed to act for R_e at $l_{e,i,s}$ so that if there are many changes in $g(n)$ we can meet the requirement at position i. n may be the only natural number for which we cannot computably bound the number of changes in $g(n)$. If we do not act at $l_{e,i,s}$ and R_e were to require attention in case (3) at some later stage t, we may never receive permission on $l_{e,i_{max}(e,t),t}$, and R_e will not be met.

Construction

Stage 0: Let $G_0 = 0^\omega$. Let $\delta_0(n) = n$ for all $n \in \omega$.

Stage s, s $\geqslant 1$:

Find the least e, if any, such that R_e requires attention at stage s. (If no such e exists, go to stage $s + 1$.) Initialize all requirements $R_{e'}$ for $e' > e$ by clearing the sequences of lengths and locations for $R_{e'}$. Choose the first case by which R_e requires attention.

If case (1) holds, let $i_{max}(e, s + 1) = 0$ and $l_{e,0,s+1}$ be fresh. Let $I(l_{e,0,s+1}) = \{0\}$.

If case (2) holds, choose the least i that is applicable. Set $G_{s+1} = f_e(x_{e,i,s}, k)\,\hat{}\,0^\omega$, where k is greatest such that $f_e(x_{e,i,s}, k)[s]\downarrow$. Clear $l_{e,i',s}$ and $x_{e,i',s}$ for all $i' > i$, and set $\delta_{s+1}(l_{e,i,s+1}) = \gamma_{s+1}(n_{max})$ where $n_{max} = \max I(l_{e,i_{max}(e,s),s})$. Set $I(l_{e,i,s+1}) = [\min I(l_{e,i,s}), n_{max}]$ and $i_{max}(e, s + 1) = i$. Say that $l_{e,i_{max}(e,s+1),s+1}$ is not waiting. We say that R_e received attention at position i at stage s.

If case (3) holds, define $i_{max}(e, s+1) = i_{max}(e, s) + 1$ and choose $l_{e,i_{max}(e,s+1),s+1}$ to be fresh. Let $I(l_{e,i_{max}(e,s+1),s+1}) = \{n\}$ where n is the least number that is not already a permitting number for some length for R_e. Say that $l_{e,i_{max}(e,s+1)-1,s+1}$ is waiting.

If case (4) holds, define $x_{e,i_{max}(e,s),s}$ to be the x found, define $i_{max}(e, s+1) = i_{max}(e, s)+1$ and choose $l_{e,i_{max}(e,s+1),s+1}$ to be fresh. Let $I(l_{e,i_{max}(e,s+1),s+1}) = \{n\}$ where n is the least number that is not already a permitting number for some length for R_e. Say that $l_{e,i_{max}(e,s+1),s+1}$ is waiting.

If we did not act in case (2) then we let $\delta_{s+1}(n) = \delta_s(n)$ for all n. If we did act in case (2), suppose we defined $\delta_{s+1}(l_{e,i,s+1})$. For all m such that $\delta_s(l_{e,i,s}) \leqslant \delta_s(m) \leqslant \delta_{s+1}(l_{e,i,s})$, we define $\delta_{s+1}(m) = \delta_{s+1}(l_{e,i,s})$. For all other m we let $\delta_{s+1}(m) = \delta_s(m)$.

End of Construction

Verification

Lemma 4.3.7. $\lim_{s<\omega} G_s$ *is well-defined and* $\lim_{s<\omega} G_s \leqslant_T A$.

Proof. We must show that for all n, $\lim_{s<\omega} \delta_s(n)$ exists. To see that $\lim_{s<\omega} \delta_s(n)$ exists, go to stage n. Note that by the way δ is defined, $\delta_s(n)$ can be changed only if there is some length $l_{e,i,s} \leqslant n$ and R_e receives attention in position i at or after stage s. So if at stage n there is no e such that $l_{e,i,n} \leqslant n$ for some i, then no requirement will be able to lift the use of $\delta(n)$ after stage n. Therefore $\delta_t(n) = \delta_n(n)$ for all $t > n$ and so the limit exists. Otherwise, choose the least e, and for this e, the least i, such that $l_{e,i,n} \leqslant n$ and R_e receives attention at position i at some stage. Let t_0 be the least stage at which R_e receives attention at position i. By the way that e and i were chosen, $l_{e,i,n}$ cannot be cleared from the sequence of lengths by the action of requirements R_i for $i < e$, or by R_e acting in a position $j < i$. Thus $\lim_s l_{e,i,s} =: l_{e,i}$ exists. R_e can receive attention in position i at most $h_e(x_{e,i,n})$ many times after stage n; R_e may be satisfied at some stage and then later require attention in case (3). If permission for $l_{e,i,n}$ is granted, R_e will receive attention in position i. We have $|\{k : f_e(x_{e,i,s},k)\downarrow\}| < h_e(x_{e,i,s})$, and so R_e can require attention in case (3) at most $h_e(x_{e,i,n})$ many times after stage n. So $\delta_s(l_{e,i})$ can change at most finitely many times. When R_e receives attention at position i at stage t_0, we will clear the sequences of lengths for requirements of lower priority, and clear $l_{e,i',n}$ for all $i' > i$. Thus if we define $l_{f,j,u}$ for either $f = e$ and $j > i$, or $f > e$, at some later stage u, we will choose fresh numbers which will be larger than n and not interfere with the approximation to $\delta(n)$. So $\lim_{s<\omega} \delta_s(n) = \delta(n)$ exists for all n.

Lastly, we must show that if $A_{s+1} \upharpoonright \delta_s(n) = A_s \upharpoonright \delta_s(n)$, then $\delta_{s+1}(n) = \delta_s(n)$ and $G_{s+1} \upharpoonright n = G_s \upharpoonright n$. If $A_{s+1} \upharpoonright \delta_s(n) = A_s \upharpoonright \delta_s(n)$ then no requirement with a length less than n can receive permission at stage s. Thus, as discussed above, $\delta_{s+1}(n) = \delta_s(n)$. Only lengths greater than n can receive permission at stage s. If a requirement R_e were to act at stage s in position i where $l_{e,i,s} > n$, we would let $G_{s+1} = f_e(x_{e,i,s},k)\,\hat{}\,0^\omega$, where k is greatest such that $f_e(x_{e,i,s},k)[s]\downarrow$. But by condition (4) of requires attention,

$f_e(x_{e,i,s}, k) \succ G_s \upharpoonright l_{e,i,s}$, and so as $l_{e,i,s} > n$, $G_{s+1} \upharpoonright n = G_s \upharpoonright n$. Therefore $\lim_{s<\omega} G_s \leqslant_T A$.

□

Let $G = \lim_{s<\omega} G_s$.

Lemma 4.3.8. *Each requirement receives attention at only finitely many stages, and is met.*

Proof. Assume by induction that there is a stage r after which no requirement R_i for $i < e$ receives attention. Suppose for contradiction that R_e receives attention at infinitely many stages. In this case we show that g is ω-c.a. by building an ω-c.a. function $\hat{g}$ with computable approximation $\langle \hat{g}_s \rangle_{s<\omega}$ such that $\lim_{s<\omega} g_s(i) = \lim_{s<\omega} \hat{g}_s(i)$ for all i.

Suppose that x is in our sequence of locations for R_e, and that x is never cleared from the sequence. We claim that we cannot have $f_e(x, k) \prec G$, where k is greatest such that $f_e(x, k) \downarrow$. Suppose for contradiction that x is in the i^{th} place in our sequence of locations at all stages after stage s. Then we cannot act for R_e at a position $j < i$, otherwise x would be cleared from the sequence. If at some stage t we have $f_e(x, k) \prec G_{t'}$ for all $t' \geqslant t$ and k is greatest such that $f_e(x, k) \downarrow$, then R_e is satisfied at all stages after stage t, and we do not act for R_e after stage t. This is a contradiction, since we act for R_e infinitely many times. Note that this shows that we must act for R_e in case (4) infinitely many times.

We now show that every natural number is contained in some permitting interval. At the first stage $s > r$ that we act for R_e, we will define $l_{e,0,s}$ and we will have $0 \in I(l_{e,0,s})$. Suppose by induction that n is in some permitting interval at stage t. As in the previous paragraph, we must act in case (4) for R_e at some later stage, and at this stage, we will define a permitting interval which includes $n + 1$.

For every n, let s_n be the stage at which a length with permitting number n is defined, and suppose this length is l_{e,i,s_n}. We define $\hat{g}$ as follows. Initially we let $\hat{g}_0(n) = g_{s_n}(n)$. Either this is the correct value of $g(n)$ and we do not need to update the approximation, or R_e receives attention at some position $j \leqslant i$ at some later stage s'. If this occurs, then we update our approximation and let $\hat{g}_{s'}(n) = g_{s'}(n)$. R_e may receive attention at position i, but this must occur at most $h_e(x_{e,i,s})$ many times. R_e may then receive attention at position $i - 1$. We will then have $n \in I(l_{e,i-1,t})$ so that changes in $g(n)$ will give permission on $l_{e,i-1,s}$, but this must occur at most $h_e(x_{e,i-1,s})$ many times, and so on. Therefore we update our approximation at most $\sum_{j \leqslant i} h_e(x_{e,j,s})$ times. Calculating the bound on the number of times the approximation $\hat{g}(n)$ may change is a computable function of n. The approximation is correct; if we updated our approximation more than $\sum_{j \leqslant i} h_e(x_{e,j,s})$ times then $f_e(x_{e,0,s}, k) \prec G$ where k is greatest such that $f_e(x_{e,0,s}, k) \downarrow$, contradicting the first claim of the lemma. Therefore we have $\lim_s \hat{g}_s(n) = \lim_s g_s(n)$ and so g is ω-c.a. This is a contradiction. Therefore R_e receives attention at only finitely many stages.

To see that R_e is satisfied, go to a stage s after which no requirement R_i for $i \leqslant e$ requires attention. Let $x = x_{e,i_{max}(e,s),s}$. If R_e is satisfied at stage s, then as R_e does not require attention in case (3) after stage s, we must have $f_e(x,k) \prec G_t$, where k is greatest such that $f_e(x,k) \downarrow$ for all $t \geqslant s$. Therefore $f_e(x,k) \prec G$ and G meets range f_e. If R_e is not satisfied at stage s, then $l_{e,i_{max}(e,s),s}$ is defined. As R_e does not require attention in case (4) after stage s, there is no extension to $G_s \upharpoonright l_{e,i_{max}(e,s),s}$ in range f_e, so G avoids range f_e, and R_e is met.

□

□

Together with the following theorem, we see that a c.e. degree computes an ω-change generic if and only if it is not totally ω-c.a.

Theorem 4.3.9. *If G is ω-change generic then G is not ω-c.a. dominated.*

Proof. Let $G = \{a_0 < a_1 < \cdots\}$. Recall that the principal function p_G of G is defined by $p_G(n) = a_n$. We show that p_G escapes domination by every ω-c.a. function.

Let f be an ω-c.a. function with computable approximation $\langle f_s \rangle_{s<\omega}$ and computable function g such that for all x, $|\{s : f_{s+1}(x) \neq f_s(x)\}| < g(x)$. Fix k. We show that there is $n \geqslant k$ such that $p_G(n) \geqslant f(n)$.

We identify strings with natural numbers via some Gödel numbering. For $\sigma \in 2^{<\omega}$, let j_σ be the number of 1s in σ. We define the partial computable function $f : \omega \times \omega \to 2^{<\omega}$. We set $f(\sigma, 0)[0] =$

$\sigma 1^{k-1} 0^{f_0(k+j_\sigma)}$ and $g'(\sigma) = g(k + j_\sigma)$. If at some later stage s we see that $f_s(k+j_\sigma) \neq f_0(k+j_\sigma)$, then we set $f(\sigma, 1)[s] = \sigma 1^{k-1} 0^{f_s(k+j_\sigma)}$. We continue in this way, defining another $f(\sigma, l)$ every time we see a change in the approximation to $f(k + j_\sigma)$. Then range f is an ω-change set of strings with computable bound g'. G cannot avoid range f as range f is dense. Therefore G must meet range f. So there is a σ such that $\sigma 1^{k-1} 0^{f(k+j_\sigma)} \prec G$. Therefore the $(k + j_\sigma)^{th}$ 1 must come after the $f(k + j_\sigma)^{th}$ bit of G. So $p_G(k + j_\sigma) \geqslant f(k + j_\sigma)$.

□

It is open whether there is a (non-c.e.) not ω-c.a. dominated degree which does not bound an ω-change generic. By Theorem 4.3.4, such a degree would have to be uniformly ω^2-c.a. dominated.

4.4 Degrees computable in generics

4.4.1 Computable in a 1-generic

In [13] and [12], Chong and Downey obtain a characterisation of the degrees which are computable in a 1-generic. We recall their characterisation. A c.e. set of strings T is a *tight cover*[1] of a set A if A neither meets nor avoids T, and for every other c.e. set of strings T' such that A neither meets nor avoids T', there is some string in T' which extends a string in T. Then a set is computable in a 1-generic set if and only if it has no c.e. tight cover.

[1]in [13] this was referred to as a Σ^0_1-dense set of strings

The proof in [13] that a set with no c.e. tight cover is computable in a 1-generic set contained a small error in the handling of Subcase 2a in the following proof. We present a proof of the theorem, including a significantly different discussion from that in [13].

Theorem 4.4.1. *Let M be a set with no c.e. tight cover. Then there is a 1-generic set G such that $M \leqslant_{\mathrm{T}} G$. In fact, $G \leqslant_{\mathrm{T}} M \oplus \varnothing''$.*

Proof. Suppose M has no c.e. tight cover. We first perform a computable construction of a Turing functional Φ. We then use the oracle $M \oplus \varnothing''$ to select a 1-generic path G through $\operatorname{dom}\Phi$ such that $\Phi(G) = M$.

Let $\langle S_e \rangle_{e<\omega}$ be an effective enumeration of all c.e. sets of strings. We must define G to meet the requirements

$$R_e : G \text{ meets or avoids } S_e.$$

Suppose that by stage s we have some string σ such that for all $m < n$, σ either meets $S_{m,s}$ or avoids S_m, and that $\Phi_s(\sigma)\downarrow \prec M$. We let G extend σ. We would now like to define Φ on some string $\sigma' \succ \sigma$ such that σ' either meets or avoids S_n, and such that $\Phi(\sigma) \prec \Phi(\sigma') \prec M$.

We wait for a string $\gamma \in S_n$ with $\gamma \succ \sigma$. If we see such a string, we can set $\Phi(\gamma) = \Phi(\sigma)$, and make G extend γ. We would like G to compute M, and so we want to define Φ on some string $\gamma'' \succ \gamma$ so that $\Phi(\gamma'')$ computes at least one more bit of M than $\Phi(\sigma)$ did. The Turing functional Φ needs to be c.e., and so in the construction of Φ, we cannot make any use of M. We therefore choose two incomparable strings $\gamma_0, \gamma_1 \succ \gamma$, and define $\Phi(\gamma_0) = \Phi(\sigma)\hat{}0$ and $\Phi(\gamma_1) = \Phi(\sigma)\hat{}1$. We say that γ_0 *guesses* that the next bit of M is 0, and that γ_1 guesses that the next bit of M is 1. Then if 0 is the next bit of M we can choose $\gamma_0 \prec G$, and if 1 is the next bit of M we can choose $\gamma_1 \prec G$.

Of course this strategy does not work with more than one requirement because there may not be any string $\gamma \succ \sigma$ in S_n. We cannot wait forever before beginning work on requirements of weaker priority.

We now consider how two strategies, one working for R_n and one working for R_{n+1}, may interact if they both work above the string σ. Suppose that at stage s_1 we see $\delta \in S_{n+1,s_1}$ with $\delta \succ \sigma$. We choose some string $\delta' \succ \delta$ such that we have made no Φ definitions on or above δ', and define $\Phi_{s_1}(\delta') = \Phi(\sigma)$. We choose two incomparable strings $\delta_0, \delta_1 \succ \delta'$, and define $\Phi_{s_1}(\delta_0) = \Phi(\sigma)\hat{}0$ and $\Phi_{s_1}(\delta_1) = \Phi(\sigma)\hat{}1$. Now suppose at some later stage s_2 we see $\gamma \in S_{n,s_2}$, and Φ is not already defined on an extension of any string β with $\sigma \preccurlyeq \beta \prec \gamma$ and $\beta \in S_{n,s_2}$. We would like to define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$, and have G extend γ' if $\Phi(\gamma') \prec M$. In order to keep Φ consistent, we look for the greatest string η with $\sigma \preccurlyeq \eta \prec \gamma$ such that $\Phi(\eta)\downarrow$. We choose some $\gamma' \succ \gamma$ such that we have made no Φ definitions on or above γ', and set $\Phi_{s_2}(\gamma') = \Phi(\eta)$. Suppose that $\gamma \succ \delta_1$. Then $\eta \succcurlyeq \delta_1$ and so $\Phi(\gamma') \succcurlyeq \Phi(\delta_1)$. However, if the next bit of M is 0, then we cannot define $\Phi(\gamma')$ for any $\gamma' \succcurlyeq \gamma$ and have $\Phi(\gamma') \prec M$.

We therefore do not want G to extend γ. We must be careful though, because S_{n+1} and S_n could repeat this situation infinitely many times and G may neither meet nor avoid S_n.

Suppose that we follow the above strategy and define Φ on a set S of extensions of strings in S_n that extend σ. If there is some $\gamma' \in S$ with $\Phi(\gamma') \prec M$, then we choose G to pass through γ', and will be able to successfully guess at the next bit of M as above. Now suppose there is no such $\gamma' \in S$. Then M does not meet $\Phi(S)$. In this situation, we would like to define Φ on some string $\tau \succ \sigma$ which avoids S_n, and such that $\Phi(\sigma) \prec \Phi(\tau) \prec M$. Then we can choose G to extend τ. We first look at the case where M in addition does not avoid $\Phi(S)$. We later consider the case where M avoids $\Phi(S)$.

Suppose M neither meets nor avoids $\Phi(S)$. We use the fact that M has no c.e. tight cover. So there is some c.e. set of strings Y that is dense in M and such that no string in Y extends a string in $\Phi(S)$. We define two strings τ_0 and τ_1 extending σ, and promise that if we define $\Phi(\rho)$ for any $\rho \succcurlyeq \tau_j$, then $\Phi(\rho)$ is an initial segment of some string in Y.

The string τ_j guesses that that the next bit of M is j. We will first wait to define $\Phi(\tau_j)$ before defining $\Phi(\rho)$ for any string $\rho \succ \tau_j$. We want to define $\Phi(\tau_j) = \Phi(\sigma)\hat{\ } j$, and so will wait for *confirmation* that our promise can be kept. If at some later stage t we see some $\nu \in Y_t$ such that $\nu \succcurlyeq \Phi(\sigma)\hat{\ } j$, then we say that τ_j is confirmed, and define $\Phi_t(\tau_j) = \Phi(\sigma)\hat{\ } j$. Once τ_j has been confirmed, we allow requirements of weaker priority than R_n to act above τ_j, as long as our promise above is kept.

We note that if we see at some stage u a string $\gamma \in S_{n,u}$ with $\sigma \preccurlyeq \gamma \prec \tau_j$, then we will choose an extension $\gamma' \succ \gamma$ that is incomparable with τ_j, regardless of whether τ_j has been confirmed by stage u. As a result, we do not define Φ on any string η with $\sigma \preccurlyeq \eta \prec \tau_j$. Therefore at any stage after we create the extension τ_j, we still wait for confirmation to define $\Phi(\tau_j) = \Phi(\sigma)\hat{\ } j$. If we do receive confirmation at some stage t, then we can define $\Phi_t(\tau_j) = \Phi(\sigma)\hat{\ } j$ and Φ will remain consistent.

As τ_j is our candidate to avoid S_n, if we do see some $\gamma \in S_n$ with $\gamma \succ \tau_j$, then we will want to define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$ in order to derive a contradiction. Therefore, if we confirm τ_j at stage t and see at some later stage u that there is $\gamma \succ \tau_j$ with $\gamma \in S_{n,u}$, then if we have not already acted for some $\beta \in S_{n,u}$ with $\tau_j \preccurlyeq \beta \prec \gamma$, we will act for γ.

Suppose j is such that $\Phi(\sigma)\hat{\ } j \prec M$. We show that τ_j avoids S_n. As Y is dense in M, there is some stage t as above where we define $\Phi_t(\tau_j) = \Phi(\sigma)\hat{\ } j$. Suppose that there is $\gamma \succcurlyeq \tau_j$ with $\gamma \in S_n$. Let $u > t$ be least such that there is $\gamma \in S_{n,u}$ with $\gamma \succcurlyeq \tau_j(\sigma)$, and for this u, let γ be the least such. By the choice of γ, we have not already acted for some $\beta \in S_{n,u}$ with $\tau_j \preccurlyeq \beta \prec \gamma$. Therefore at stage u we define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$. Let η be greatest such that $\eta \prec \gamma$ and $\eta \in \operatorname{dom} \Phi_{u-1}$. Then we set $\Phi(\gamma') = \Phi(\eta)$. As $\tau_j \in \operatorname{dom} \Phi_{u-1}$ and $\gamma \succcurlyeq \tau_j$, we must have $\eta \succcurlyeq \tau_j$. We have kept the promise that if we define $\Phi(\rho)$ for strings $\rho \succcurlyeq \tau_j$, then $\Phi(\rho)$ is an initial segment of some string in Y. So $\Phi(\gamma') = \Phi(\eta)$ is an initial segment of some string in Y, which is a contradiction. We can then let G extend τ_j.

One problem with the strategy to avoid S_n is that we do not know during the construction of Φ what the index for the c.e. set Y is. For notational convenience, we write Y_i for the i^{th} c.e. set of strings when considering strings in the range of Φ. We instead define at stage s infinitely many pairwise incomparable strings $\tau_{i,j}(\sigma)$ for $i < \omega$ and $j < 2$. The string $\tau_{i,j}(\sigma)$ guesses that i is an index of Y, and that the next bit of M is j. We refer to the strings $\tau_{i,j}(\sigma)$ as *extensions*. If we wish to define $\Phi(\rho) = \mu$ for any string $\rho \succcurlyeq \tau_{i,j}(\sigma)$, then we will wait for confirmation. In this case, we now wait until a stage t where we see a string $\nu \in Y_{i,t}$ such that $\nu \succcurlyeq \mu$. We may then define $\Phi_t(\rho) = \mu$. We wait until an extension $\tau_{i,j}(\sigma)$ is confirmed before allowing requirements of weaker priority than R_n to act above $\tau_{i,j}(\sigma)$.

Suppose that at stage s we have defined the extension $\tau_{i,j}(\sigma)$, but have not yet confirmed it. If there is some $\gamma \in S_{n,s}$ with $\gamma \succcurlyeq \tau_{i,j}(\sigma)$, then we will now need to wait for $\tau_{i,j}(\sigma)$ to be confirmed before we can define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$. This will do no harm to the construction. If i is such that Y_i is dense in M, and j is such that $\Phi(\sigma)\,\hat{}\,j \prec M$, then we will confirm $\tau_{i,j}(\sigma)$ at some stage $t > s$. Then at stage t we will immediately define $\Phi(\gamma') = \Phi(\tau_{i,j}(\sigma))$ for some $\gamma' \succ \gamma$, and as $\Phi(\tau_{i,j}(\sigma)) = \Phi(\sigma)\,\hat{}\,j \prec M$, we will have $\Phi(\gamma') \prec M$ as desired.

Suppose we confirm the extension $\tau_{i',j}(\sigma)$ at stage t. We allow

R_{n+1} to start working above $\tau_{i',j}(\sigma)$. We may have made several promises about defining Φ on strings extending σ, for the sake of requirements of stronger priority than R_n. Suppose that I is the set of indices such that if we want to define $\Phi(\rho) = \mu$ for some $\rho \succ \sigma$, then we require for every $i \in I$ some string $\nu \in Y_{i,t'}$ such that $\nu \succcurlyeq \mu$. We associate the set I with σ by *tagging* the string σ with the set I, and write $I(\sigma) = I$. In order to define $\Phi(\rho) = \mu$ for a string extending $\rho \succ \tau_{i',j}(\sigma)$, we must in addition see that there is some string $\nu \in Y_{i',t'}$ such that $\nu \succcurlyeq \mu$. Therefore we tag $\tau_{i',j}(\sigma)$ with the set $I(\tau_{i',j}(\sigma)) = I(\sigma) \cup \{i'\}$.

Above the string $\tau_{i',j}(\sigma)$, we are guessing that if M does not meet or avoid $\Phi(S)$, then i' is an index for the set Y. We may define Φ on strings above $\tau_{i',j}(\sigma)$. If i' is not an index for the set Y, then the argument given above that $\tau_{i',j}(\sigma)$ avoids S_n does not work. So there may be strings $\gamma \in S_n$ with $\gamma \succcurlyeq \tau_{i',j}(\sigma)$. Suppose we see such a γ at stage $u > t$. We wish to define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$ as usual, in the hopes that $\Phi(\gamma') \prec M$. We do define $\Phi_u(\gamma')$ as usual, and choose extensions $\gamma_0, \gamma_1 \succ \gamma'$. We wait for confirmation to define $\Phi(\gamma_j) = \Phi(\gamma')\hat{}\,j$, and set $I(\gamma_j) = I(\sigma)$. If Y_i is dense in M for all $i \in I(\gamma_j)$ and j is the next bit of M, then we will eventually confirm γ_j and define $\Phi(\gamma_j)$.

If we define $\Phi(\rho)$ for any $\rho \succ \gamma_j$, we must have $\Phi(\rho) \succcurlyeq \Phi(\gamma_j)$. Therefore, if $\Phi(\gamma_j) \not\prec M$, there is no point in pursuing a strategy to define $\Phi(\rho)$ for a string $\rho \in S_n$ with $\rho \succ \gamma_j$. We do not know during the construction whether $\Phi(\gamma_j) \prec M$, so above γ_j, we work under the assumption that $\Phi(\gamma_j) \prec M$.

Strategies of weaker priority than R_n may have acted above $\tau_{i',j}(\sigma)$ after stage t, but before the stage u where we defined $\Phi(\gamma')$. At stage u, we no longer allow requirements of weaker priority than R_n to work above any string that extends $\tau_{i',j}(\sigma)$. We remove the tag $I(\tau_{i',j}(\sigma))$ from $\tau_{i',j}(\sigma)$. If we confirm γ_j at some later stage v, then we allow R_{n+1} to work above γ_j at stage v. As in a standard finite injury priority argument, we eventually come to some string τ' such that R_{n+1} works above τ' from some stage on.

If $\Phi(\gamma_j) \not\prec M$, then the strategy to avoid S_n will continue to be in place. That is, all other extensions $\tau_{i,j}(\sigma)$ will still have their tag, and we allow requirements of weaker priority to act above $\tau_{i,j}(\sigma)$, as long as we continue to enforce the promises we must keep above each such extension. If i^* is an index for Y and j is such that $\Phi(\sigma)\hat{\ }j \prec M$, then we will confirm $\tau_{i^*,j}(\sigma)$ at some stage and define $\Phi(\tau_{i^*,j}(\sigma))$. Then we never remove the tag of $\tau_{i^*,j}(\sigma)$, $\tau_{i^*,j}(\sigma)$ will avoid S_n, and we will have $\Phi(\sigma) \prec \Phi(\tau_{i^*,j}(\sigma)) \prec M$.

We finally consider the case where M does not meet $\Phi(S)$, but M avoids $\Phi(S)$. Then there is some $\mu \succ \Phi(\sigma)$ such that $\mu \prec M$, and for all $\gamma' \in S$, $\Phi(\gamma') \not\succcurlyeq \mu$. We would like to define Φ on some string $\tau \succ \sigma$ such that $\Phi(\tau) = \mu$. Then τ will avoid S_n.

In order to define such a τ, we will need to again guess at the next bits of M. Along with the extensions $\tau_{i,j}(\sigma)$, we create two special extensions of σ, which we denote $\tau_{\emptyset,j}(\sigma)$ for $j < 2$. We wait for confirmation to define $\Phi(\tau_{\emptyset,j}(\sigma)) = \Phi(\sigma)\hat{\ }j$. We tag $\tau_{\emptyset,j}(\sigma)$

with the set $I(\tau_{\varnothing,j}(\sigma)) = I(\sigma)$, and so if Y_i is dense in M for all $i \in I(\sigma)$ and j is the next bit of M, then we will eventually confirm $\tau_{\varnothing,j}(\sigma)$ and define $\Phi(\tau_{\varnothing,j}(\sigma))$ at some stage t. Once $\tau_1 = \tau_{\varnothing,j}(\sigma)$ has been confirmed, we create the extensions $\tau_{i,j}(\tau_1)$ for $i \in \omega \cup \{\varnothing\}$. We allow only R_n to work above $\tau_{\varnothing,j}(\tau_1)$. If some $\tau_{i,j}(\tau_1)$ for $i \in \omega$ is confirmed at some later stage, then we allow R_{n+1} to work above $\tau_{i,j}(\tau_1)$. By guessing the bits $j_1, \dots, j_m$ such that $\mu = \Phi(\sigma)\hat{\ }j_1\hat{\ }\dots\hat{\ }j_m$, we eventually come at some stage u to the string τ as above.

As with the other extensions $\tau_{i,j}(\sigma)$ for $i \in \omega$, $\tau_{\varnothing,j}(\sigma)$ is our candidate to avoid S_n. So if we do see some $\gamma \in S_n$ with $\gamma \succ \tau_{\varnothing,j}(\sigma)$, then we will want to define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$ in order to derive a contradiction. Therefore, if we confirm $\tau_{\varnothing,j}(\sigma)$ at stage u and see at some later stage v that there is $\gamma \succ \tau_{\varnothing,j}(\sigma)$ with $\gamma \in S_{n,v}$, then if we have not already acted for some $\beta \in S_{n,v}$ with $\tau_{\varnothing,j}(\sigma) \preccurlyeq \beta \prec \gamma$, we will act for γ.

We show that τ avoids S_n. Suppose that there is $\gamma \succcurlyeq \tau$ with $\gamma \in S_n$. Let $u > t$ be least such that there is $\gamma \in S_{n,u}$ with $\gamma \succcurlyeq \tau_{i^*,j}(\sigma)$, and for this u, let γ be the least such. By the choice of γ, we have not already acted for some $\beta \in S_{n,u}$ with $\tau \preccurlyeq \beta \prec \gamma$. Therefore at stage u we define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$. Let η be greatest such that $\eta \prec \gamma$ and $\eta \in \operatorname{dom}\Phi_{u-1}$. Then we set $\Phi(\gamma') = \Phi(\eta)$. As $\tau \in \operatorname{dom}\Phi_{u-1}$ and $\gamma \succcurlyeq \tau$, we must have $\eta \succcurlyeq \tau$. Then $\Phi(\gamma') = \Phi(\eta) \succcurlyeq \Phi(\tau) = \mu$, which is a contradiction. We now give the formal details of the construction.

Construction

Stage 0: Define $\Phi(\lambda) = \lambda$, where λ is the empty string. Let $I(\lambda) = \varnothing$, and say that R_0 is able to work above λ.

Stage s, $s \geqslant 1$:

Step 1: *defining* Φ *on extensions of strings in* S_n.

There is an $n \in \omega$ and strings σ and γ such that

1. $\sigma \in \operatorname{dom} \Phi_{s-1}$,
2. R_n is able to work above σ,
3. $\gamma \in S_{n,s}$ with $\gamma \succ \sigma$, and
4. (a) if $\gamma \succ \tau_{i,j}(\sigma)$ for some extension $\tau_{i,j}(\sigma)$, then R_n has not already acted for any string β with $\tau_{i,j}(\sigma) \prec \beta \prec \gamma$ and $\beta \in S_{n,s}$, or
   (b) if γ is not comparable with any extension $\tau_{i,j}(\sigma)$, or $\gamma \prec \tau_{i,j}(\sigma)$ for some extension $\tau_{i,j}(\sigma)$, then R_n has not already acted for any string β with $\sigma \prec \beta \prec \gamma$ and $\beta \in S_{n,s}$.

Let n be least such that there is σ and γ as above. For this n, let σ be such that there is γ as above, and choose γ least.

If there is some extension $\tau_{i,j}(\sigma)$ with $\sigma \prec \tau_{i,j}(\sigma) \preccurlyeq \gamma$, and $\tau_{i,j}(\sigma)$ has not been confirmed, then we proceed to the next step.

Otherwise, let $\eta \preccurlyeq \gamma$ be greatest with $\eta \in \operatorname{dom} \Phi_{s-1}$. Choose a string $\gamma' \succ \gamma$ incomparable with all strings extending γ generated in the construction so far. Define $\Phi_s(\gamma') = \Phi(\eta)$. We choose incomparable strings $\gamma_0, \gamma_1 \succ \gamma'$. We add γ_0 and γ_1 to L. For $j < 2$, we say that R_n is waiting for confirmation to define $\Phi(\gamma_j) = \Phi(\gamma')\hat{\ } j$ at stage s, and tag γ_j with the set $I(\gamma_j) = I(\sigma)$.

For all strings $\mu \in \operatorname{dom} \Phi_{s-1}$ with either $\sigma \prec \mu \prec \gamma$ or $\mu \succ \gamma$, if $l > n$ and R_l was able to work above μ at stage s, then R_l is unable to work above μ at stage $s + 1$, we remove the tag of μ, and we remove all extensions defined for μ from L_{s-1}.

We say that R_n acts for the string γ.

Step 2: *confirmation.*

For every string $\rho \in L$ at the end of Step 1, we do the following. If R_n is waiting for confirmation to define $\Phi(\rho) = \mu$, and for all $i \in I(\rho)$ there is $\nu \in Y_{i,s}$ such that $\nu \succcurlyeq \mu$, then we declare ρ as confirmed. We remove ρ from L, and define $\Phi_s(\rho) = \mu$. If ρ is not of the form $\tau_{\varnothing,j}(\sigma)$, then we say that R_{n+1} is able to work above ρ at stage $s + 1$. If ρ is of the form $\tau_{\varnothing,j}(\sigma)$, then we say that R_n is able to work above ρ at stage $s + 1$.

Step 3: *creating extensions.*

For any string σ that is maximal in the domain of Φ at the end of Step 2, we do the following. Suppose that R_n is able to work above σ at the end of Step 2. If σ already has its extensions defined, we proceed to the next stage. Otherwise, we create infinitely many pairwise incomparable extensions $\tau_{i,j}(\sigma)$ of σ, for $i \in \omega \cup \{\varnothing\}$ and $j < 2$, so that $\tau_{i,j}(\sigma)$ is incomparable with any string that extends σ defined during the construction so far. We say that σ has had its extensions defined.

For $i \in \omega$, we tag $\tau_{i,j}(\sigma)$ with the set $I(\tau_{i,j}(\sigma)) = I(\sigma) \cup \{i\}$, and we tag $\tau_{\emptyset,j}(\sigma)$ with the set $I(\tau_{\emptyset,j}(\sigma)) = I(\sigma)$. We say that R_n is waiting for confirmation to define $\Phi(\tau_{i,j}(\sigma)) = \Phi(\sigma)\hat{}\, j$ at stage s.

For each $\tau_{i,j}(\sigma)$ we have just created, we perform Step 2. That is, if for all for all $i \in I(\tau_{i,j}(\sigma))$ there is $\nu \in Y_{i,s}$ such that $\nu \succcurlyeq \Phi(\sigma)\hat{}\, j$, we declare $\tau_{i,j}(\sigma)$ as confirmed, remove $\tau_{i,j}(\sigma)$ from L, and define $\Phi_s(\tau_{i,j}(\sigma)) = \Phi(\sigma)\hat{}\, j$. If $i \in \omega$, then we say that R_{n+1} is able to work above $\tau_{i,j}(\sigma)$ at stage $s + 1$.

End of Construction

We now define $G \leqslant_{\mathrm{T}} M \oplus \emptyset''$ such that $\Phi(G) = M$. We do this in *Steps* by finite extension.

At Step 0, we set $G_0 = \lambda$. We proceed by induction and assume that by Step n we have $G_n = \sigma$ for some $\sigma \in \operatorname{dom} \Phi_{t_n}$, where $t = t_n$ is a stage in the construction such that

1. R_n is able to work above σ at all stages after stage t,
2. σ has tag $I(\sigma)$ at all stages after stage t,
3. for all $i \in I(\sigma)$, Y_i is dense in M,
4. $|\Phi(\sigma)| \geqslant n$ and $\Phi(\sigma) \prec M$, and
5. for all $m < n$, σ either meets S_{m,t_m} or avoids S_m.

Suppose at some stage $s > t$ we define $\Phi_s(\gamma')$ in Step 1 of the construction for some $\gamma' \succ \gamma$ where $\gamma \in S_n$ and $\gamma \succ \sigma$. Let S be the set of all such γ'.

Case 1: M meets $\Phi(S)$.

Choose $\gamma \in S_n$ with $\gamma \succ \sigma$ least such that there is $\gamma' \succ \gamma$ with $\gamma' \in S$ and $\Phi(\gamma') \prec M$. Suppose that $\Phi(\gamma')$ is defined at stage $u \geqslant t$. By induction, we cannot act for R_m for $m < n$ after stage u. By part (4) of the conditions in Step 1 and the choice of γ, we do not act for R_n for any string in S_n that is comparable with γ after stage u. At stage u we choose strings $\gamma_0, \gamma_1 \succ \gamma'$, and R_n waits for confirmation. We tag γ_j with the set $I(\gamma_j) = I(\sigma)$, and by assumption, Y_i is dense in M for all $i \in I(\sigma)$. Therefore if j is such that $\Phi(\gamma')\hat{}\,j \prec M$, we will confirm γ_j at some later stage v, and define $\Phi_v(\gamma_j) = \Phi(\gamma')\hat{}\,j$. We say that R_{n+1} is able to work above γ_j at stage v. Again by induction, R_{n+1} is able to work above γ_j at all stages $v' \geqslant v$, and γ_j has tag $I(\gamma_j)$ at all stages $v' \geqslant v$. We have $G_{n+1} = \gamma_j$.

Case 2: M does not meet $\Phi(S)$. There are two subcases.

Subcase a: M avoids $\Phi(S)$. Then there exists some $\mu \succ \Phi(\sigma)$ such that $\mu \prec M$ and for all $\gamma' \in S$, $\Phi(\gamma') \nsucc \mu$. Suppose $\mu = \Phi(\sigma)\hat{}\,j_1\hat{}\ldots\hat{}\,j_m$.

Consider the string $\tau_1 = \tau_{\emptyset,j_1}(\sigma)$. We have $I(\tau_1) = I(\sigma)$, and by assumption, Y_i is dense in M for all $i \in I(\sigma)$. So at some later stage u_1 we will confirm τ_1 and define $\Phi_{u_1}(\tau_1) = \Phi(\sigma)\hat{}\,j_1$. At stage u_1 we define the extensions $\tau_{i,j}(\tau_1)$ for $i \in \omega \cup \{\emptyset\}$ and $j < 2$. Again, at some later stage u_2 we will confirm $\tau_2 = \tau_{\emptyset,j_2}(\tau_1)$ and define $\Phi_{u_2}(\tau_2) = \Phi(\sigma)\hat{}\,j_1\hat{}\,j_2$. We eventually come at some stage u_m to a string τ_m where $\Phi_{u_m}(\tau_m) = \mu$.

We show that τ_m avoids S_n. So suppose for contradiction that there is some $\gamma \in S_n$ with $\gamma \succcurlyeq \tau_m$. Let $v > u_m$ be least such that there is $\gamma \in S_{n,v}$ with $\gamma \succcurlyeq \tau_m$, and for this v, let γ be the least such. By the choice of γ, we have not already acted for some $\beta \in S_{n,u}$ with $\tau_m \preccurlyeq \beta \prec \gamma$. Then at stage v we will define $\Phi_v(\gamma')$ for some $\gamma' \succ \gamma$. Let η be greatest such that $\eta \prec \gamma$ and $\eta \in \operatorname{dom} \Phi_{u-1}$. Then we set $\Phi(\gamma') = \Phi(\eta)$. As $\tau_m \in \operatorname{dom} \Phi_{v-1}$ and $\gamma \succcurlyeq \tau_m$, we must have $\eta \succcurlyeq \tau_m$. As $\gamma' \succ \eta \succcurlyeq \tau_m$ and Φ is consistent, $\Phi(\gamma') = \Phi(\eta) \succcurlyeq \Phi(\tau_m) = \mu$, which contradicts the choice of μ. So τ_m avoids S_n.

At stage u_m we create the extensions $\tau_{i,j}(\tau_m)$ for $i \in \omega \cup \{\varnothing\}$ and $j < 2$. Let i' be such that $Y_{i'}$ is dense in M, and let j' be such that $\mu\hat{\ } j' \prec M$. We have $I(\tau_{i',j'}(\tau_m)) = I(\sigma) \cup \{i'\}$, and so at some later stage w we will confirm $\tau_{i',j'}(\tau_m)$ and define $\Phi_w(\tau_{i',j'}(\tau_m)) = \mu\hat{\ } j'$. At stage w we say that R_{n+1} is able to work above $\tau_{i',j'}(\tau_m)$. By induction and the fact that $\tau_{i',j'}(\tau_m)$ avoids S_n, this is true at all stages after stage w, and $\tau_{i',j'}(\tau_m)$ has tag $I(\tau_{i',j'}(\tau_m))$ at all stages after stage w. We have $G_{n+1} = \tau_{i',j'}(\tau_m)$.

Subcase b: M does not avoid $\Phi(S)$. As M does not have a c.e. tight set of strings, there is an i^* such that Y_{i^*} is dense in M and such that for all $\alpha \in Y_{i^*}$ and $\beta \in \Phi(S)$, $\alpha \not\succcurlyeq \beta$.

We note the following crucial fact. Suppose the extension τ is confirmed at some stage u. Then at any stage $v > u$, if we define $\Phi_v(\rho)$ for some string $\rho \succcurlyeq \tau$ and R_n has not already acted for some string comparable with τ, then ρ was confirmed at stage v, and $I(\rho) \supseteq I(\tau)$.

Let j be such that $\Phi(\sigma)\hat{\ } j \prec M$. At stage t we create the extensions $\tau_{i,j}(\sigma)$, and R_n waits for confirmation to define $\Phi(\tau_{i,j}(\sigma)) = \Phi(\sigma)\hat{}j$. Let $u \geqslant t$ be least such that for all $i \in I(\sigma) \cup \{i^*\}$, there is some $v \in Y_{i,u}$ such that $v \succcurlyeq \Phi(\sigma)\hat{}j$. Then at stage u we declare the string $\tau_{i^*,j}(\sigma)$ as confirmed, and define $\Phi_u(\tau_{i^*,j}(\sigma)) = \Phi(\sigma)\hat{}j$.

To show that $\tau_{i^*,j}(\sigma)$ avoids S_n, suppose for contradiction that there is $\gamma \in S_n$ with $\gamma \succcurlyeq \tau_{i^*,j}(\sigma)$. Let $v > u$ be least such that there is $\gamma \in S_{n,v}$ with $\gamma \succcurlyeq \tau_{i^*,j}(\sigma)$, and for this v, let γ be the least such. By the choice of v, we have not already acted for some $\beta \in S_{n,v}$ with $\tau_{i^*,j}(\sigma) \preccurlyeq \beta \prec \gamma$. Therefore at stage v we will define $\Phi(\gamma')$ for some $\gamma' \succ \gamma$. Let η be greatest such that $\eta \prec \gamma$ and $\eta \in \operatorname{dom} \Phi_{v-1}$. We set $\Phi(\gamma') = \Phi(\eta)$. As $\tau_{i^*,j}(\sigma) \in \operatorname{dom} \Phi_{v-1}$ and $\gamma \succcurlyeq \tau_{i^*,j}(\sigma)$, we must have $\eta \succcurlyeq \tau_{i^*,j}(\sigma)$. Suppose $\Phi(\eta)$ was defined at stage w. Again by the choice of γ, and the fact above, η was confirmed at stage w, and $I(\eta) \supseteq I(\tau_{i^*,j})$. In particular, $i^* \in I(\eta)$, and so there is some $v \in Y_{i^*,w}$ such that $v \succcurlyeq \Phi(\eta)$. So $v \succcurlyeq \Phi(\gamma')$, which contradicts the choice of i^*. So $\tau_{i^*,j}(\sigma)$ avoids S_n. We have $G_{n+1} = \tau_{i^*,j}(\sigma)$.

□

4.4.2 Computable in an ω-change generic

We now consider the analogue of the above result for ω-change generics. We say that range f is a *ω-change tight cover* of a set A if range f is an ω-change set of strings, A neither meets nor avoids range f, and for all other ω-change sets of strings range f', if A neither meets nor avoids range f', then there is some string in range f' which extends a string in range f.

Suppose that M does not have an ω-change tight cover. Then if range f is an ω-change set of strings and M neither meets nor avoids range f, then there is a partial computable function $y : \omega \times \omega \to 2^{<\omega}$ and a partial computable function $w : \omega \to \omega$ such that range y is an ω-change set of strings with partial computable bound w, and no string in range y extends a string in range f. The difference between M not having a c.e. tight cover and not having an ω-change tight cover is that the existence of y and w is guaranteed even if M meets range f as a c.e. set of strings. So there may be some x and k such that $f(x,k)\downarrow< M$, but if M does not meet range f *as an ω-change set of strings*, that is, there is $k' > k$ such that $f(x,k')\downarrow\not< M$, then such a y and w must still exist.

Theorem 4.4.2. *Let M be a set with no ω-change tight cover. Then there is an ω-change generic set G such that $M \leqslant_{\mathrm{T}} G$. In fact, $G \leqslant_{\mathrm{T}} M \oplus \varnothing''$.*

Proof. The proof will follow fairly closely the proof of the previous theorem. Suppose M has no ω-change tight cover. We first perform a computable construction of a Turing functional Φ. We then use the oracle $M \oplus \varnothing''$ to select an ω-change generic path G through dom Φ such that $\Phi(G) = M$.

Let $\langle f_i, h_i\rangle_{i<\omega}$ be an effective list such that $\langle$range $f_i\rangle$ is a list of all ω-change sets of strings, and that range f_i has partial computable bound h_i. We must meet the requirements

$$R_e : G \text{ meets or avoids range } f_e.$$

Suppose that by stage s we have some string σ such that for all $m < n$, σ either meets range f_m or avoids range f_m, and that $\Phi_s(\sigma)\downarrow \prec M$. We let G extend σ. We would now like to define Φ on some string $\sigma' \succ \sigma$ such that σ' either meets or avoids range f_n, and such that $\Phi(\sigma) \prec \Phi(\sigma') \prec M$.

The strategy to define $\Phi(\sigma')$ will be similar to the strategy from the previous proof. Suppose at stage s_1 we see some x with $f_n(x, 0)[s_1]\downarrow \succ \sigma$, and $h_n(x)[s_1]\downarrow$. Then we would like to choose some $\gamma' \succ f_n(x, 0)$, define $\Phi(\gamma')$, and have G extend γ' if $f_n(x, 1)\uparrow$ and $\Phi(\gamma') \prec M$. If β is greatest with $\beta \preccurlyeq f_n(x, 0)$ and $\beta \in \operatorname{dom}\Phi_{s_1-1}$, then we choose some string $\gamma' \succ f_n(x, 0)$ such that no definition for Φ has been made on or above γ', and define $\Phi_{s_1}(\gamma') = \Phi(\beta)$. We choose two incomparable extensions $\gamma_0, \gamma_1 \succ \gamma'$, and wait for confirmation to define $\Phi(\gamma_j) = \Phi(\gamma')\,\hat{}\,j$ as before. Suppose we confirm γ_j at some later stage s_2. Then above γ_j we assume that $\Phi(\gamma_j) \prec M$, and so we do not need to act again for R_n above γ_j unless we see at some later stage s_3 that $f(x, 1)[s_3]\downarrow$. We allow R_{n+1} to work above γ_j at stage $s_2 + 1$.

Now suppose we do see at some later stage s_3 that $f_n(x, 1)[s_3]\downarrow\succ f_n(x, 0)$. Then we would like to define $\Phi(\gamma'')$ for some $\gamma'' \succ f_n(x, 1)$, and have G extend γ'' if $f_n(x, 2)\uparrow$ and $\Phi(\gamma'') \prec M$. We repeat the above for $f_n(x, 1)$. Strategies working for requirements of weaker priority than R_n may have started to work above strings extending $f_n(x, 0)$. We restart all such strategies above $f_n(x, 1)$. As

$$|\{k : f_n(x, k)\downarrow\}| < h_n(x),$$

we eventually no longer need to act for R_n, and will define Φ on some string extending $f_n(x,k)$ where k is such that $f_n(x,k') \uparrow$ for all $k' > k$.

Suppose that we follow the above strategy and define Φ on a set $S \subset$ range f_n of strings extending σ. Then we show that $\Phi(S)$ is an ω-change set of strings. We set $\bar{h} = h$, and if we choose some $\gamma' \succ f_n(x,k)$ and define $\Phi(\gamma')$ at stage s, then we set $\bar{f}_n(x,k) = \Phi(\gamma')$.

Suppose M meets the ω-change set of strings range $\bar{f}_n = \Phi(S)$. Then if γ' is such that $\Phi(\gamma') \prec M$ and $\gamma' \succ f_n(x,k)$, then we let G extend γ'. As M meets range $\bar{f}_n$, there is no $k' > k$ such that $f_n(x,k') \downarrow$, and so G meets range f_n. We will be able to successfully guess at the next bit of M. Now suppose that M does not meet range $\bar{f}_n$. As in the previous proof, there are two cases in this situation. If M avoids range $\bar{f}_n$, then we will be able to show that there is a string $\tau \succ \sigma$ such that τ avoids range f_n. If M does not avoid range $\bar{f}_n$, then we use the fact that M has no ω-change tight cover. In this case we will also be able to show that there is a string $\tau \succ \sigma$ such that τ avoids range f_n.

Consider the case where M neither meets nor avoids range $\bar{f}_n$. As M has no ω-change tight cover, there is a partial computable function y with partial computable bound w such that range y is an ω-change set, and no string in range y extends a string in range $\bar{f}_n$. Let $\langle y_i, w_i \rangle$ be an effective list such that $\langle \text{range}\, y_i \rangle$ is a list of all ω-change sets of strings, and that range y_i has partial computable bound w_i. We create extensions $\tau_{i,j}(\sigma)$ with the intention that $\tau_{i,j}(\sigma)$

guesses that i is an index for the pair $\langle y, w\rangle$ in the above list, and that j is the next bit of M. The extensions $\tau_{i,j}(\sigma)$ function in the same way as in the previous proof.

Because M avoids range $\bar{f}_n$ as an ω-change set of strings if and only if it avoids range $\bar{f}_n$ as a c.e. set of strings, the cases where M does not meet $\bar{f}_n$ are identical to the cases where M does not meet $\Phi(S)$ in the previous proof.

Construction

Stage 0: Define $\Phi(\lambda) = \lambda$, where λ is the empty string. Let $I(\lambda) = \varnothing$, and say that R_0 is able to work above λ.

Stage s, $s \geqslant 1$:

Step 1: *defining* Φ *on extensions of strings in range* f_n.

There is an $n \in \omega$, a string σ, and $x, k \in \omega$ such that

1. $\sigma \in \operatorname{dom} \Phi_{s-1}$,
2. R_n is able to work above σ,
3. $f_n(x, k)[s]\downarrow$ and $f_n(x, k) \succ \sigma$,
4. (a) if $f_n(x, k) \succ \tau_{i,j}(\sigma)$ for some extension $\tau_{i,j}(\sigma)$, and either
      i. R_n has not already acted for any string β with $\tau_{i,j}(\sigma) \prec \beta \prec f_n(x, k)$ and $\beta \in$ range $f_{n,s}$, or
      ii. R_n has acted for the string $f_n(x, k-1)$ but not for $f_n(x, k)$.

   (b) if $f_n(x, k)$ is not comparable with any extension $\tau_{i,j}(\sigma)$, or $f_n(x, k) \prec \tau_{i,j}(\sigma)$ for some extension $\tau_{i,j}(\sigma)$, and either

i. R_n has not already acted for any string β with $\sigma \prec \beta \prec f_n(x,k)$ and $\beta \in \operatorname{range} f_{n,s}$, or

ii. R_n has acted for the string $f_n(x,k-1)$ but not for $f_n(x,k)$.

Let n be least such that there are σ, x, and k as above. For this n, let σ be such that there is x as above, and choose x least.

If there is some extension $\tau_{i,j}(\sigma)$ with $\sigma \prec \tau_{i,j}(\sigma) \preccurlyeq f_n(x,k)$, and $\tau_{i,j}(\sigma)$ has not been confirmed, then we proceed to the next step.

Otherwise, let $\eta \preccurlyeq f_n(x,k)$ be greatest with $\eta \in \operatorname{dom} \Phi_{s-1}$. Choose a string $\gamma' \succ f_n(x,k)$ incomparable with all strings extending $f_n(x,k)$ generated in the construction so far. Define $\Phi_s(\gamma') = \Phi(\eta)$. We choose incomparable strings $\gamma_0, \gamma_1 \succ \gamma'$. We add γ_0 and γ_1 to L. For $j < 2$, we say that R_n is waiting for confirmation to define $\Phi(\gamma_j) = \Phi(\gamma')\hat{\ }j$ at stage s, and tag γ_j with the set $I(\gamma_j) = I(\sigma)$.

For all strings $\mu \in \operatorname{dom} \Phi_{s-1}$ with either $\sigma \prec \mu \prec f_n(x,k)$ or $\mu \succ f_n(x,k)$, if $l > n$ and R_l was able to work above μ at stage s, then R_l is unable to work above μ at stage $s+1$, we remove the tag of μ, and we remove all extensions defined for μ from L_{s-1}.

We say that R_n acts for the string $f_n(x,k)$.

Step 2: *confirmation.*

For every string $\rho \in L$ at the end of Step 1, we do the following. If R_n is waiting for confirmation to define $\Phi(\rho) = \mu$, and for all $i \in I(\rho)$ there is $\nu \in \operatorname{range} y_{i,s}$ such that $\nu \succcurlyeq \mu$, then we declare ρ as confirmed. We remove ρ from L, and define $\Phi_s(\rho) = \mu$. If ρ is not of the form $\tau_{\varnothing,j}(\sigma)$, then we say that R_{n+1} is able to work above ρ

at stage $s + 1$. If ρ is of the form $\tau_{\emptyset,j}(\sigma)$, then we say that R_n is able to work above ρ at stage $s + 1$.

Step 3: *creating extensions.*

For any string σ that is maximal in the domain of Φ at the end of Step 2, we do the following. Suppose that R_n is able to work above σ at the end of Step 2. If σ already has its extensions defined, we proceed to the next stage. Otherwise, we create infinitely many pairwise incomparable extensions $\tau_{i,j}(\sigma)$ of σ, for $i \in \omega \cup \{\emptyset\}$ and $j < 2$, so that $\tau_{i,j}(\sigma)$ is incomparable with any string that extends σ defined during the construction so far. We say that σ has had its extensions defined.

For $i \in \omega$, we tag $\tau_{i,j}(\sigma)$ with the set $I(\tau_{i,j}(\sigma)) = I(\sigma) \cup \{i\}$, and we tag $\tau_{\emptyset,j}(\sigma)$ with the set $I(\tau_{\emptyset,j}(\sigma)) = I(\sigma)$. We say that R_n is waiting for confirmation to define $\Phi(\tau_{i,j}(\sigma)) = \Phi(\sigma)\hat{\ } j$ at stage s.

For each $\tau_{i,j}(\sigma)$ we have just created, we perform Step 2. That is, if for all for all $i \in I(\tau_{i,j}(\sigma))$ there is $\nu \in$ range $y_{i,s}$ such that $\nu \succcurlyeq \Phi(\sigma)\hat{\ } j$, we declare $\tau_{i,j}(\sigma)$ as confirmed, remove $\tau_{i,j}(\sigma)$ from L, and define $\Phi_s(\tau_{i,j}(\sigma)) = \Phi(\sigma)\hat{\ } j$. If $i \in \omega$, then we say that R_{n+1} is able to work above $\tau_{i,j}(\sigma)$ at stage $s + 1$.

End of Construction

$\square$

4.5 Downward density of generics in the Δ^0_2 degrees

4.5.1 Downward density of 1-generics

In [14], Chong and Jockusch showed that the 1-generic degrees below $\emptyset'$ are downward dense. That is, if $A \leqslant_T \emptyset'$ is 1-generic and $B \leqslant_T A$ is noncomputable, then there is $C \leqslant_T B$ that is 1-generic. This was extended by Haught ([24]) who showed that if $A \leqslant_T \emptyset'$ is 1-generic and $B \leqslant_T A$ is noncomputable, then there is $C \equiv_T B$ that is 1-generic. We give another proof of Haught's theorem.

We say that a partial function v from strings to strings is an *extension* function if $v(\alpha) \geqslant \alpha$ for every $\alpha \in \operatorname{dom} v$. A set is 1-generic if it meets or avoids every c.e. set of strings. Equivalently, a set is 1-generic if it meets or avoids the range of every partial computable extension function.

Theorem 4.5.1 (Haught [24])**.** *Let $\boldsymbol{a}$ be a Δ^0_2 1-generic Turing degree. Then every noncomputable degree below $\boldsymbol{a}$ is 1-generic.*

Proof. Let $A \in \boldsymbol{a}$ be 1-generic with computable approximation $\langle A_s \rangle$. Suppose that Φ is a Turing functional such that $\Phi(A)$ is noncomputable. We build Turing functionals Γ and Δ, such that $\Gamma(\Phi(A))$ is 1-generic and $\Delta(\Gamma(\Phi(A))) = \Phi(A)$.

Let $\langle S_e \rangle_{e<\omega}$ be an effective enumeration of all c.e. sets of strings. We must meet the requirements

$$R_e : \Gamma(\Phi(A)) \text{ meets or avoids } S_e.$$

We first discuss how the definition of Γ and the strategies to meet the requirements interact. We discuss the definition of Δ later. We write $\sigma \prec \tau$ to denote that σ is a proper initial segment of τ. For a string ρ, let the *sibling* of ρ be the string $\hat{\rho}$ of the same length as ρ, and which differs from ρ on only its last bit.

Suppose that at stage s we have have a string $\alpha \prec A_s$ with $\alpha \prec A_t$ for all $t \geqslant s$ (and so $\alpha \prec A$) such that $\Gamma(\Phi(\alpha))$ meets or avoids S_d for all $d < e$. To meet R_e, we look for strings $\sigma \in S_e$ with $\sigma \succ \Gamma(\Phi(\alpha))$. We may never see such a string, and so we will work towards satisfying requirements of weaker priority. Suppose that at stage s_1 we see $\mu \in S_{f,s_1}$ with $\mu \succ \Gamma(\Phi(\alpha))$, for some $f > e$. We then define $\Gamma_{s_1}(\Phi(\alpha')) = \mu$ for some $\alpha' \prec A_{s_1}$ with $\Phi(\alpha') \succ \Phi(\alpha)$.

At stage $s_2 > s_1$ we see $\sigma \in S_{e,s_2}$ with $\sigma \succ \Gamma(\Phi(\alpha))$. If $\Phi(A)[s_2] \succ \Phi(\alpha')$, then $\Gamma(\Phi(A))[s_2] \succ \mu$. Therefore if σ is incomparable with μ, $\Gamma(\Phi(A))[s_2]$ does not extend σ. We will want $\Phi(A)$ to later become incomparable with $\Phi(\alpha')$ so that we may define $\Gamma(\Phi(A))$ to extend σ. We however have no way of ensuring this occurs. We may have $\Phi(\alpha') \prec \Phi(A)[t]$ for all $t \geqslant s_2$, and so $\Gamma(\Phi(A))$ does not extend σ. This is of no concern to us at stage s_2, but if we act as above infinitely many times for requirements of weaker priority than R_e, only to later see a string enter S_e that $\Gamma(\Phi(A))$ does not extend, then $\Gamma(\Phi(A))$ will neither meet nor avoid S_e.

We therefore need some way of forcing $\Phi(A)$ to change. We make use of the totality and noncomputability of $\Phi(A)$, as well as the 1-genericity of A. We will want a configuration $\Phi(A_t)$ of $\Phi(A)$ that extends $\Phi(\alpha)$ but is incomparable with $\Phi(\alpha')$. Then if at some stage we see a string $\sigma \in S_e$ that extends $\Gamma(\Phi(A))$ but is incomparable with μ, we can use the 1-genericity of A to force A to extend A_t, which in turn forces $\Phi(A)$ to extend $\Phi(A_t)$, and allows us to define $\Gamma(\Phi(A))$ to extend σ.

As $\Phi(A)$ is total and noncomputable, if $\alpha \prec A$, then there are infinitely many Φ-splits above α. Therefore, if at stage t_1 we see $\mu \in S_{f,t_1}$ with $\mu \succ \Gamma(\Phi(\alpha))$, before defining Γ on any string extending $\Phi(\alpha)$, we wait until a stage $t_2 \geqslant t_1$ where we find Φ-splits τ_1, τ_2 above α. We are then free to set $\Gamma(\Phi(\tau_2)) = \mu$. Then if at stage t_3 we see $\sigma \in S_{e,t_3}$ with σ incomparable with $\Gamma(\Phi(\tau_2))$, we start to define the partial computable extension function v_e. We set $v_e(\alpha) = \tau_1$. We also define $\Gamma(\Phi(v_e(\alpha))) = \sigma$. This puts pressure on A to change to extend $v_e(\alpha)$, so that $\Phi(A)$ extends $\Phi(v_e(\alpha))$, and $\Gamma(\Phi(A))$ meets S_e.

Of course, A does not need to extend $v_e(\alpha)$ for it to be 1-generic. However, if we act as above infinitely many times and define $v_e(\alpha)$ for infinitely many α, then range v_e will be dense in A. As A is 1-generic, it will have to extend some $v_e(\alpha)$, which means that $\Gamma(\Phi(A))$ must meet S_e.

We are keeping the string $\Phi(\tau_1)$ in *reserve* for R_e, in case we see some stage t_3 and some string $\sigma \in S_{e,t_3}$ with $\sigma \succ \Gamma(\Phi(\alpha))$ but incomparable with $\Gamma(\Phi(\tau_2))$. Suppose that at stage $u_2 > t_2$ we have $\Phi(A)[u_2] \succ \Phi(\tau_1)$, but there is no string $\sigma \in S_{e,u_2}$ with $\sigma \succ \Gamma(\Phi(\alpha))$. We are not ready to define Γ on any string extending $\Phi(\tau_1)$, because we may run into the same trouble as before where we had no way of forcing $\Phi(A)$ to change later if necessary. If $\Phi(A)[t] \succ \Phi(\tau_1)$ for all $t \geqslant u_2$, then we must define Γ on some string extending $\Phi(\tau_1)$ so that $\Gamma(\Phi(A))$ is total. However, if this is the case, then $\Phi(\tau_1) \prec \Phi(A)$. As A is Δ^0_2, there is some stage $u_3 \geqslant u_2$ and a string ϕ with $\phi \prec A_u$ for all $u \geqslant u_3$ such that $\Phi(\phi) \succcurlyeq \Phi(\tau_1)$. Then as ϕ is an initial segment of A, at some later stage u_4 we will find Φ-splits τ_3, τ_4 above ϕ such that $\Phi(\tau_3), \Phi(\tau_4) \succ \Phi(\tau_1)$. If $A_{u_4} \succ \tau_4$, then we can define $\Gamma(\Phi(\tau_4))$ however we wish, as long as Γ is kept consistent, and keep the string $\Phi(\tau_3)$ in reserve, as we did with $\Phi(\tau_1)$ before. Lemma 4.5.3 below uses the 1-genericity of A to show that this cannot repeat infinitely many times, and we eventually come to a final string we keep in reserve for R_e.

We will let $\Gamma(\Phi(\tau_4)) = \Gamma(\Phi(\tau_2))$. If we do see at some stage $t_4 > u_4$ a string $\sigma \in S_{e,t_4}$ that is incomparable with $\Gamma(\Phi(\tau_2))$, then we can still define $v_e(\phi) = \tau_3$ as an attempt to force A to change, so that we may define $\Gamma(\Phi(A))$ to extend σ.

More generally, suppose that at stage s, $\gamma \prec \Phi_s(A_s)$ is maximal in $\operatorname{dom}\Gamma_{s-1}$, and that we would like to define $\Gamma(\gamma') = \mu$ for some $\gamma' \succ \gamma$ with $\gamma' \prec \Phi_s(A_s)$ in order to satisify R_f. Then we must ensure that for all $e < f$, if μ does not meet $S_{e,s}$, then there is some string, which we write as $\mathrm{res}_{\gamma,e,s}$, which extends γ but is incomparable with γ', which we keep in reserve for R_e. This means that we will need to find many Φ-splits at stage s before we define $\Gamma(\gamma') = \mu$. This causes us no concern, as there are infinitely many Φ-splits above any initial segment of A.

The last situation we must consider in the definition of Γ is the following. Suppose that at stage s, $\gamma \prec \Phi_s(A_s)$ is the greatest string such that $\Gamma_{s-1}(\gamma)\downarrow$. If $\Phi_s(A_s)$ is incomparable with all $\mathrm{res}_{\gamma',i,s-1}$ for all $\gamma' \succ \gamma$ with $\gamma' \in \operatorname{dom}\Gamma_{s-1}$ and all i, then we can define $\Gamma(\Phi_s(A_s))$ however we would like, as long as Γ is kept consistent. The most progress that we have made above $\Gamma(\gamma)$ is the last string that we enumerated into the range of Γ which extends $\Gamma(\gamma)$. So we let $\Gamma(\Phi_s(A_s))$ be that string.

We now discuss the definition of Δ. Suppose that we have α and s as above, and we have defined $\nu = \Delta(\Gamma(\Phi(\alpha))) \preccurlyeq \Phi(\alpha) \prec \Phi(A)$. We see the string $\mu \in S_{f,s_1}$ at stage s_1, find the Φ-splits τ_1, τ_2 above α at stage s_2, and if $A_{s_2} \succ \tau_2$, define $\Gamma(\Phi(\tau_2)) = \mu$. We will want to define Δ on some string extending $\Gamma(\Phi(\alpha))$ so that we can compute more of $\Phi(A)$. We must make sure that Δ is consistent, and that $\Delta(\Gamma(\Phi(A)))$ correctly computes $\Phi(A)$. We ask that if $\mu \prec \Gamma(\Phi(A))$, then $\Delta(\mu)$ computes the next bit of $\Phi(A)$ after ν. We therefore have to *guess* at stage s_2 what the next bit of $\Phi(A)$ will be. We have to guess in such a way that if we are wrong, then we are able to correct ourselves later on.

If $\Phi_{s_2}(A_{s_2}) \succcurlyeq \Phi(\tau_2) \succ \nu\hat{\ } j$, then our guess at stage s_2 is that j is the next bit of $\Phi(A)$. If we knew that our guess were correct, we could define $\Delta(\mu) = \nu\hat{\ } j$. However, even if our guess is wrong, we might still like $\Gamma(\Phi(A))$ to extend μ in order to meet R_f. But then $\Delta(\mu)$ is wrong about $\Phi(A)$. We instead define $\Gamma_{t_1}(\Phi(\tau_2)) = \mu\hat{\ } j$ and $\Delta(\mu\hat{\ } j) = \nu\hat{\ } j$.

Now suppose that at stage s_3 we see that $\Phi(A)[s_3] \succ \Phi(\alpha)$, but is incomparable with both $\Phi(\tau_1)$ and $\Phi(\tau_2)$. If $\Phi(A)[s_3] \succ \nu\hat{\ }(1-j)$, we guess at stage s_3 that $1-j$ is the next bit of $\Phi(A)$. We would like $\Gamma(\Phi(A))[s_3]$ to still extend μ to satisfy R_f, and so define $\Gamma(\Phi(A))[s_3] = \mu\hat{\ }(1-j)$ and $\Delta(\mu\hat{\ }(1-j)) = \nu\hat{\ }(1-j)$.

Now suppose at stage s_4 we have $\Phi(A)[s_4] \succ \Phi(A)[s_3]$, but see $\sigma \in S_{e,s_4}$ that is incomparable with with $\Gamma(\Phi(A))[s_3] = \mu\hat{\ }(1-j)$. If $\sigma \succ \mu\hat{\ }j$, then we would not want $\Gamma(\Phi(A))$ to extend $\mu\hat{\ }j$, because at stage s_4 we are guessing that $(1-j)$ is the next bit of $\Phi(A)$, but $\Delta(\mu\hat{\ }j) = \nu\hat{\ }j$. The trouble is that we have defined $\Gamma(\Phi(A))[s_3]$ and $\Gamma(\Phi(\tau_2))$ to be siblings.

We need to ensure that if we define $\Gamma(\Phi(\tau)) = \rho$ at stage t, then for every string σ incomparable with ρ, there is some $\sigma' \succ \sigma$ where $\Delta_{t-1}(\sigma')\uparrow$. Then if we see a change in $\Phi(A)$ that allows us to define $\Gamma(\Phi(A))$ to extend σ, we can define $\Gamma(\Phi(A))$ to extend σ' and let $\Delta_t(\sigma')$ guess however it wishes about the next bit of $\Phi(A)$. Therefore, the domain of Δ, and so the range of Γ, must be *sparse*. We will define Γ so that at every stage t, range Γ_t does not contain both a string and its sibling. Then there will be some σ' as above. We now turn to the formal details of the construction. For convenience, we let $\Theta = \Gamma \circ \Phi$.

Construction

Stage 0: Define $\Gamma(\lambda) = \lambda$ and $\Delta(\lambda) = \lambda$, where λ is the empty string.

Stage $s, s \geqslant 1$:

Step 1: defining Γ *and* Δ.

Let $\gamma \prec \Phi_s(A_s)$ be greatest such that $\Gamma_{s-1}(\gamma)\downarrow$, and let $\alpha \prec A_s$ be such that $\Phi_s(\alpha) = \gamma$.

1. γ is not maximal in $\operatorname{dom}\Gamma_{s-1}$.

   (a) There is some e such that $\mathrm{res}_{\gamma,e,s-1}\downarrow$ and $\Phi_s(A_s) \succ \mathrm{res}_{\gamma,e,s-1}$. If $\Gamma_{s-1}(\mathrm{res}_{\gamma,e,s-1})\downarrow$, then we proceed to the next step. Otherwise, if there are Φ-splits τ_1, τ_2 above α such that $\Phi(\tau_1), \Phi(\tau_2) \succ \mathrm{res}_{\gamma,e,s-1}$, if $A_s \succ \tau_2$, we set $\mathrm{res}_{\gamma,e,s} = \Phi(\tau_1)$.

   (b) There is some β with $\alpha \prec \beta \preccurlyeq A_s$ such that $\Phi(\beta)$ is incomparable with γ' for all $\gamma' \in \operatorname{dom}\Gamma_{s-1}$ with $\gamma' \succ \gamma$, and $\mathrm{res}_{\gamma,e,s-1}$ for all e. Let β be the least such.

   Then with $\tau = \tau_2$ in subcase (a), or $\tau = \beta$ in subcase (b), we do the following. Let f be least such that for some $\gamma' \in \operatorname{dom}\Gamma_{s-1}$ with $\gamma' \succ \gamma$, $\Gamma(\gamma')$ was defined for R_f. Let $\sigma \in S_{f,s}$. Choose some $\sigma' \succ \sigma$ such that the sibling of σ' is not in range Γ_{s-1}. Define $\Gamma(\Phi(\tau)) = \sigma'$. Let j be such that $\Phi(\tau) \succ \Delta(\Gamma(\Phi(\alpha)))\hat{\ }j$. Define $\Delta(\sigma') = \Delta(\Gamma(\Phi(\alpha)))\hat{\ }j$.

2. γ is maximal in $\operatorname{dom}\Gamma_{s-1}$, and there is $e \leqslant s$ such that $\Gamma(\gamma)$ does not meet $S_{e,s}$, but there is $\sigma \in S_{e,s}$ with $\sigma \succ \Gamma(\gamma)$. (We choose the least such e, and for this e, the least such σ.) Suppose $i_1, \ldots, i_n$ are the indices $i < e$ such that σ does not meet $S_{i,s}$.

   If there are $n+1$ many Φ-splits $\rho_1, \ldots, \rho_{n+1}$ at stage s above α, then we choose some string $\sigma' \succ \sigma$ such that the sibling of σ'

is not in range Γ_{s-1}. Assume $A_s \succcurlyeq \rho_{n+1}$. Define $\Gamma_s(\Phi(\rho_{n+1})) = \sigma'$ and set $\mathrm{res}_{\gamma,i_k,s} = \Phi(\rho_k)$ for all $k = 1, \ldots, n$. We say that $\Gamma(\Phi(\rho_{n+1}))$ was defined for R_e. Let j be such that $\Phi(\rho_{n+1}) \succ \Delta(\Gamma(\Phi(\alpha)))\hat{}j$. Define $\Delta(\sigma') = \Delta(\Gamma(\Phi(\alpha)))\hat{}j$. We set $v_e(\alpha) = \rho_{n+1}$.

Step 2: defining v_e.

For each $\gamma' \prec \Phi_s(A_s)$ with $\gamma' \in \operatorname{dom}\Gamma_s$, and $e \leqslant s$, we do the following. Let $\gamma \prec \gamma'$ be greatest such that $\gamma \in \operatorname{dom}\Gamma_s$. If $\Gamma(\gamma')$ was defined for R_f at stage $t \leqslant s$ and $f > e$, and if there is $\sigma \in S_{e,s}$ such that $\sigma \succ \Gamma(\gamma)$ but $\sigma \not\prec \Gamma(\gamma')$, we do the following. Suppose τ and $u \leqslant s$ are such that $\Phi(\tau) = \mathrm{res}_{\gamma,e,s}$ with $\tau \preccurlyeq A_u$. If $\alpha \prec \tau$ is such that $\Phi(\alpha) = \gamma$, then set $v_e(\alpha) = \tau$. Choose some $\sigma' \succ \sigma$ such that the sibling of σ' is not in range Γ_{s-1}. Define $\Gamma_s(\Phi(v_e(\alpha))) = \sigma'$. Let j be such that $\Phi(v_e(\alpha)) \succ \Delta(\Gamma(\Phi(\alpha)))\hat{}\,j$. Define $\Delta(\sigma') = \Delta(\Gamma(\Phi(\alpha)))\hat{}j$. We say that $\Gamma(\Phi(v_e(\alpha))$ was defined for R_e.

End of Construction

It is immediate that Γ and Δ are consistent. We isolate the following simple lemma, which will be used in the remaining parts of the verification.

Lemma 4.5.2. *Suppose $\langle A_s \rangle$ is a computable approximation of a Δ^0_2 set A, and Φ is a Turing functional. Let T be an infinite c.e. set of strings which are initial segments of some A_s. Then T is dense in A.*

Proof. As T is as infinite set of binary strings, it contains string of arbitrary length. Furthermore, for any length l and any stage s, T must contain a string of length at least l which was enumerated after stage s.

Let $\sigma \prec A$, and let s be such that $\sigma \prec A_t$ for all $t \geqslant s$. We show that there is $\tau \in T$ with $\tau \succ \sigma$. Let τ be a string of length greater than $|\sigma|$ which was enumerated into T after stage s. Then $\tau \succ \sigma$. □

Lemma 4.5.3. *Suppose that $\Gamma(\gamma)$ was defined at stage s. For $t \geqslant s$ and $e \in \omega$, let $T_{\gamma,e,t}$ be the set of strings $\gamma' \in \operatorname{dom}\Gamma_t$ such that $\gamma' \succ \gamma$, $\Gamma(\gamma')$ was defined for R_f with $f > e$ at stage $t' \leqslant t$, but $\Gamma(\gamma')$ does not meet $S_{e,t'}$. Then if $T_{\gamma,e,t}$ is nonempty, there is a string $\mathrm{res}_{\gamma,e,t} \succ \gamma$ such that*

1. $\mathrm{res}_{\gamma,e,t} = \Phi(\tau)$ *for some $\tau \preccurlyeq A_u$ and some $u \leqslant t$,*
2. *if $\Gamma_t(\xi)\downarrow$ for any $\xi \succcurlyeq \mathrm{res}_{\gamma,e,t}$, then $\Gamma_t(\mathrm{res}_{\gamma,e,t})\downarrow$ and meets $S_{e,t}$,*
3. $\mathrm{res}_{\gamma,e,t}$ *is incomparable with γ' for all $\gamma' \in T_{\gamma,e,t}$, and*
4. $\{\,\mathrm{res}_{\gamma,e,t} : \mathrm{res}_{\gamma,e,t}\downarrow\,\}$ *is a set of pairwise incomparable strings.*

Furthermore, if $T_{\gamma,e,t}$ is nonempty at some stage $t \geqslant s$, then $\lim_u \mathrm{res}_{\gamma,e,v}$ *exists. That is, there is some v such that* $\mathrm{res}_{\gamma,e,v'} = \mathrm{res}_{\gamma,e,v}$ *for all $v' \geqslant v$.*

Proof. For the first part of the lemma, we suppose by induction that the lemma holds for stage $s-1$, and analyse the different subcases of the construction. As in the construction, let $\gamma \prec \Phi_s(A_s)$ be greatest such that $\gamma \in \operatorname{dom}\Gamma_{s-1}$, and α be such that $\Phi(\alpha) = \gamma$.

It is clear that the result holds if we define $\Gamma(\gamma')$ at stage s via Case 2 of Step 1, or via Step 2. Suppose that we act in Subcase 1(a) at stage s, $\Phi_s(A_s) \succ \mathrm{res}_{\gamma,e,s-1}$, and we find the Φ-splits τ_1, τ_2 above α. We set $\mathrm{res}_{\gamma,e,s} = \Phi(\tau_1)$, so point 1 of the lemma holds, and we define $\Phi(\tau_2)$. Then $\mathrm{res}_{\gamma,e,s}$ is incomparable with $\Phi(\tau_2)$ because τ_1 and τ_2 are Φ-splits. For all $\mathrm{res}_{\gamma,i,s-1}\downarrow$ with $i \neq e$, $\mathrm{res}_{\gamma,i,s} = \mathrm{res}_{\gamma,i,s-1}$, and point 4 holds by the inductive assumption. Finally suppose that we act in Subcase 1(b) at stage s. Then we define $\Gamma(\Phi(\beta))$, where $\Phi(\beta)$ is incomparable with γ' for all $\gamma' \in \operatorname{dom}\Gamma_{s-1}$ with $\gamma' \succ \gamma$, and $\mathrm{res}_{\gamma,e,s-1}$ for all e. We have $\mathrm{res}_{\gamma,e,s-1} = \mathrm{res}_{\gamma,e,s}$ for all e, and the result holds at stage s by the inductive assumption.

For the second part of the lemma, suppose for contradiction that $\lim_u \mathrm{res}_{\gamma,e,u}$ does not exist. Then there is an infinite sequence $t_1, t_2, \ldots$ of stages where $\mathrm{res}_{\gamma,e,t_{j-1}} \neq \mathrm{res}_{\gamma,e,t_j}$. We have $\mathrm{res}_{\gamma,e,t_j} \succ \mathrm{res}_{\gamma,e,t_{j-1}}$ for all j. We redefine $\mathrm{res}_{\gamma,e,s}$ at a stage s where $\Phi_s(A_s) \succ \mathrm{res}_{\gamma,e,s-1}$. Suppose at stage t_j we find the splits τ_{1,t_j} and τ_{2,t_j} above $\alpha_{t_j} \prec A_{t_j}$. Then $\mathrm{res}_{\gamma,e,t_j} = \Phi(\tau_{1,t_j})$. For all j, $\Phi_t(A_t) \succ \Phi(\tau_{1,t_j})$ for infinitely many t, and so $\Phi(A) \succ \Phi(\tau_{1,t_j})$. As τ_{1,t_j} and τ_{2,t_j} are Φ-splits, $\Phi(\tau_{1,t_j})$ and $\Phi(\tau_{2,t_j})$ are incomparable, and so $\Phi(A) \not\succ \Phi(\tau_{2,t_j})$.

The set $T = \{\, \tau_{2,t_j} \mid j \in \omega \,\}$ is an infinite c.e. set of strings which are initial segments of some A_s. By Lemma 4.5.2, T is dense in A. As A is 1-generic, $A \succ \tau_{2,t_j}$ for some j. Therefore $\Phi(A) \succ \Phi(\tau_{2,t_j})$, contradicting our observation above.

□

Lemma 4.5.4. *The range of* Γ *is infinite.*

Proof. Suppose for contradiction that range Γ is finite, and that we do not enumerate any new strings into range Γ after stage s_1. We cannot act in Step 2 after stage s_1 because we would then enumerate a new string into range Γ.

Suppose $s_2 \geqslant s_1$ is such that $\mathrm{res}_{\gamma,e,s}$ has reached its limit by stage s_2 for all $\gamma \in \operatorname{dom}\Gamma_{s_1}$ and e. Then we cannot act in Case 1(a) after stage s_2.

Consider the set

$$C = \operatorname{dom}\Gamma_{s_2} \cup \{\, \mathrm{res}_{\gamma,e,s_2} : \gamma \in \operatorname{dom}\Gamma_{s_2}, \mathrm{res}_{\gamma,e,s_2}\downarrow \}.$$

Let $C' = \{\hat{\rho} : \rho \preccurlyeq \nu, \nu \in C\}$. Then by the way that β is chosen in Subcase 1(b) of the construction, $\operatorname{dom}\Gamma \subseteq C'$. As C' is finite, $\operatorname{dom}\Gamma$ is finite too. Let $s_3 \geqslant s_2$ be such that $\operatorname{dom}\Gamma = \operatorname{dom}\Gamma_{s_3}$.

As $\operatorname{dom}\Gamma$ is finite and $\Phi(A)$ is total, let γ be the maximal string in $\operatorname{dom}\Gamma$ such that $\Phi(A) \succ \gamma$. As we do not act in Case 1 after stage s_3, γ is maximal in $\operatorname{dom}\Gamma$. Let α be such that $\gamma = \Phi(\alpha)$, and suppose that $s_4 \geqslant s_3$ is such that $\alpha \prec A_s$ for all $s \geqslant s_4$.

Let $s_5 \geqslant s_4$ be least such that there is $e \leqslant s_5$ with $\sigma \in S_{e,s_5}$, $\sigma \succ \Gamma(\gamma)$, $i_1, \ldots, i_n$ are the indices $i < e$ such that σ does not meet S_{i,s_5}, and there are $n + 1$ many Φ-splits above α. Such a stage exists because there are infinitely many indices e such that S_e contains a string extending $\Gamma(\gamma)$, and there are infinitely many Φ-splits above any initial segment of A. Then at stage s_5 we will enumerate an axiom into Γ. This is a contradiction, and so the range of Γ is infinite.

□

Lemma 4.5.5. $\Gamma(\Phi(A))$ *is total.*

Proof. Let $L_{\geqslant k} = \{\alpha : |\Gamma(\Phi((\alpha))| \geqslant k\}$. By Lemma 4.5.4, range Γ is infinite, and so contains arbitrarily long strings. Therefore for any k, $L_{\geqslant k}$ is an infinite c.e. set of strings which are initial segments of some A_s. By Lemma 4.5.2, $L_{\geqslant k}$ is dense in A. As A is 1-generic, A meets $L_{\geqslant k}$, and $\Gamma(\Phi(A))$ is total.

□

Lemma 4.5.6. *Each requirement is met, and so* $\Gamma(\Phi(A))$ *is 1-generic.*

Proof. Suppose for contradiction that $\Gamma(\Phi(A))$ neither meets nor avoids S_e.

If $v_e(\alpha) \prec A$ for some α, then $\Gamma(\Phi(A))$ meets S_e. As $\Gamma(\Phi(A))$ is total and $\Gamma(\Phi(A))$ neither meets nor avoids S_e, we act in Step 2 of the construction at infinitely many stages, and so define $v_e(\alpha)$ for infinitely many strings α. Then range v_e is an infinite c.e. set of strings. By Lemma 4.5.2, range v_e is dense in A. As A is 1-generic, there is α such that $v_e(\alpha) \prec A$. Then $\Gamma(\Phi(A))$ meets S_e, and this is a contradiction.

□

Lemma 4.5.7. $\Delta(\Gamma(\Phi(A))) = \Phi(A)$.

Proof. Suppose that we have $\alpha \prec A$, and $\Delta(\Gamma(\Phi(\alpha))) \prec \Phi(A)$. Let $\alpha' \prec A$ with $\alpha' \succ \alpha$ be such that $\Gamma(\Phi(\alpha'))$ is the least initial segment of $\Gamma(\Phi(A))$ extending $\Gamma(\Phi(\alpha))$. Such an α' exists because $\Gamma(\Phi(A))$ is total.

Suppose that $\gamma' \in \operatorname{dom}\Gamma$, and that $\gamma \in \operatorname{dom}\Gamma$ is greatest with $\gamma \prec \gamma'$. Let j be $\gamma'(|\gamma|)$. Then $\Delta(\Gamma(\gamma')) = \Delta(\Gamma(\gamma))\hat{\ } j$.

As $\alpha' \prec A$, $\Phi(\alpha') \prec \Phi(A)$. Let j be $\Phi(\alpha')(|\Phi(\alpha)|)$. Then we have $\Delta(\Gamma(\Phi(\alpha'))) = \Delta(\Gamma(\Phi(\alpha)))\hat{\ } j$. So $\Delta(\Gamma(\Phi(\alpha'))) \succ \Delta(\Gamma(\Phi(\alpha)))$ and $\Delta(\Gamma(\Phi(\alpha'))) \prec \Phi(A)$.

□

□

4.5.2 Failure of downward density for ω-change generics

Downward density for pb-generics was shown to fail by Schaeffer in [40]. He constructed a pb-generic A below $\varnothing'$ that bounds a non-computable array computable set B. As all pb-generics must be array noncomputable and array noncomputability is upwards closed, we see that B cannot bound a pb-generic set.

Martin (see [26]) showed that downward density holds for 2-generics, so we might wonder whether downward density can be recovered when we move to the stronger genericity notion of ω-change genericity. We conjecture that this fails, and give a sketch of what seems to be the most difficult part of the proof. We hope to confirm this in future work.

Conjecture 4.5.8. *Downward density for ω-change generics below $\varnothing'$ fails.*

Sketch of proof.

We construct a Δ_2^0 set A which is ω-change generic, and a Turing functional Φ such that $\Phi(A)$ is noncomputable and does not bound an ω-change generic set. We therefore meet the requirements

$$P_e : A \text{ is } \omega\text{-change generic}$$

$$N_e : \Phi(A) \text{ is noncomputable}$$

$$Q_e : \text{if } \Gamma_e(\Phi(A)) \text{ is total, then } \Gamma_e(\Phi(A)) \text{ is not } \omega\text{-change generic}$$

where $\langle \Gamma_e \rangle$ is an effective list of all Turing functionals.

The strategy to meet P_e is familiar. We choose a length p and then look for extensions to $A_s \restriction p$ in range f_e. If we see at stage s' some x and k such that $f_e(x,k)[s'] \succ A_{s'} \restriction p$, we say that the strategy is realised at stage s'. We ensure that $f_e(x,k') \prec A$, where k' is greatest such that $f_e(x,k')\downarrow$. We know that after stage s', we must act at most $h_e(x)$ many times in order to satisfy P_e.

The strategy to meet N_e is also familiar. We choose a length n and wait until a stage s where φ_e, the eth partial computable function, successfully computes the first n bits of $\Phi(A)[s]$. In this case we say that the strategy has been realised. We then change $\Phi(A)$ on its nth bit to diagonalise against φ_e.

In order to meet Q_e, if $\Gamma_e(\Phi(A))$ is total, then we must build an ω-change set of strings T such that $\Gamma_e(\Phi(A))$ neither meets nor avoids T. As such a set T must be infinite, we see that action for this type of requirement may be infinitary. We place the construction on a tree of strategies. A strategy working for a Q-requirement will have two outcomes. The infinite outcome guesses that $\Gamma_e(\Phi(A))$ is total, and therefore that we must act infinitely many times to build the ω-change set of strings T. The finite outcome guesses that $\Gamma_e(\Phi(A))$ is not total, and so no further action is required to meet the requirement after some stage.

The strategy τ for Q_e will pick some length q and will wait for Γ_e to give a definition on some string extending $\Phi(A)[s] \upharpoonright q$. If Γ_e never gives such a definition, then we keep $\Phi(A)[s] \upharpoonright q$ as an initial segment of $\Phi(A)$. Then Γ_e is not total on $\Phi(A)$, which gives us a global win on Q_e. If we do see such a definition at some stage s, then we will begin to define the ω-change set of strings T which we would like for $\Gamma_e(\Phi(A))$ to neither meet nor avoid. We pick some new x and define $g(x,0)$ to be some string incomparable with $\Gamma_e(\Phi(A)[s] \upharpoonright q)$.

We must also declare a value for $h(x)$. The length q might be quite long. In particular, there may be many lengths n with $n < q$ belonging to N-requirements of weaker priority than τ that have not yet been realised. We let $h(x)$ be twice the number of such N-

requirements. This is unavoidable, since there must be infinitely many strategies working for N-requirements below the node τ.

Suppose that at some later stage t we see that an N-strategy of weaker priority than τ with length $n < q$ is realised. We must create a split in $\Phi(A)$, and so a split in A, and redirect A through this split. Suppose that we create the split σ in A and let A extend σ. We continue the construction with A above σ, which will inevitably involve creating further splits in A above σ. This situation is dangerous for us. Splits in $\Phi(A)$ are beneficial for our opponent. Suppose we have defined $\Phi(\rho)$ for some string ρ with $\rho \succ \sigma$, but Γ_e has not been defined on any string equal to or extending $\Phi(\rho)$. Then he is free to define $\Gamma_e(\Phi(\rho)) = g(x, 0)$. He can then press the genericity of A to force A to pass through ρ, in which case $\Phi(A)$ must pass through $\Phi(\rho)$, so that $\Gamma_e(\Phi(A))$ extends $g(x, 0)$. Then $\Gamma_e(\Phi(A))$ will meet our set T.

Suppose that μ is a strategy working for a P-requirement, and that μ lies below the infinite outcome of τ. Further suppose that μ is realised at some stage u, where we see $f_e(x, k)[u] \downarrow\succ \sigma$. There may be many splits ρ that we have created in A with $\rho \succcurlyeq f_e(x, k)$, and for which Γ_e has given no definition for $\Gamma_e(\Phi(\rho))$. If we allow the multiple genericity of A to force us to extend such splits, then as above, Γ_e can give a definition which will mean that $\Gamma_e(\Phi(A))$ meets our set T.

At stage u, we will instead redirect A through a string ν_1 which is incomparable with σ. If ρ_1 was a split in A with $\rho_1 \succcurlyeq f_e(x, k)$ for which no definition $\Gamma_e(\Phi(\rho_1))[u]$ has been made, then we choose some extension $\nu' \succ \nu_1$, and define $\Phi(\nu') = \Phi(\rho_1)$. We force the opponent to make a Γ_e definition on a string extending $\Phi(\rho_1)$. If he never makes such a definition, then we let A extend ν_1 forever, so that Γ_e is not total on $\Phi(A)$. This gives us a global win on τ. As μ lies below the infinite outcome of τ, this particular strategy for meeting P_e is abandoned, but some strategy for P_e below the finite outcome of τ can ensure that P_e is still met. Similarly, we can attempt to meet the requirement N_e under the finite outcome of τ. If Γ_e does give us such a definition on some string $\gamma \succcurlyeq \Phi(\rho_1)$, we know that if $\Phi(A)$ extends γ, $\Gamma_e(\Phi(A))$ must extend $\Gamma_e(\gamma)$.

Note that it is of no benefit to our opponent if he defines Γ_e on some string properly extending $\Phi(\rho)$. If he does define Γ_e on $\gamma \succ \Phi(\rho)$ and uses the genericity of A to make A pass through ρ, we are only required to have $\Phi(A)$ pass through $\Phi(\rho)$. We can therefore let A pass through ρ and choose $\Phi(A)$ to pass through some string extending $\Phi(\rho)$ but not γ. Then we have satisfied the commitment to the genericity of A, but do not need to worry about $\Gamma_e(\Phi(A))$ extending $g(x,0)$. We therefore assume that if ρ is a split, and so $\Phi(\rho)$ is defined, that if Γ_e gives a definition on some string extending $\Phi(\rho)$, it gives a definition on $\Phi(\rho)$.

We repeat the actions in the previous paragraph for every split ρ_i for which $\rho_i \succcurlyeq f_e(x,k)$ at stage u. If Γ_e demonstrates that it is total in each case, we know all of the definitions $\Gamma_e(\Phi(\rho_i))$ that it can make. Once we have seen all such definitions, we let $g(x,1)$

be a string that is incomparable with $\Gamma_e(\Phi(\rho_i))$ for all i, and can then let A extend $f_e(x,k)$. If $f_e(x,k)$ is equal to some ρ_i, then we are currently meeting the genericity requirement of A, and $\Gamma_e(\Phi(A))$ does not extend $g(x,1)$. In this case, if we later see that $f_e(x,k+1)\downarrow$, then $f_e(x,k+1) \succcurlyeq f_e(x,k)$, and so letting A extend $f_e(x,k+1)$ will do no harm since $\Gamma_e(\Phi(A))$ will still not extend $g(x,1)$.

Otherwise, if at some later stage v we see $f_e(x,k+1)[v]\downarrow$, then $f_e(x,k+1)$ might either extend a split ρ_i which was above $f_e(x,k)$ at stage u, or there may be further splits ρ' extending $f_e(x,k+1)$ that were defined after we let A extend $f_e(x,k)$, but before stage v. In the former case, we can let A extend $f_e(x,k+1)$, and as in the previous paragraph, $\Gamma_e(\Phi(A))$ will not extend $g(x,1)$. In the latter case, a split that we create after we let A extend $f_e(x,k)$ but before stage v must either extend a split that was there at stage u, or be created for the sake of a weaker priority N-requirement with length $n < q$. We then may be forced to extend such a split, in which case $\Gamma_e(\Phi(A))$ might extend $g(x,1)$, but there are only $h(x)$ many such N-requirements. Therefore we may repeat the above strategies, and will reach some stage where we satisfy the genericity of A and ensure that $\Gamma_e(\Phi(A))$ does not extend $g(x,k)$, for some k with $k < 2h(x)$.

Bibliography

[1] Ambos-Spies, K., Kjos-Hanssen, B., Lempp, S., and Slaman, T. A. Comparing DNR and WWKL. *J. Symbolic Logic 69*, 4 (2004), 1089–1104.

[2] Barmpalias, G., Day, A., and Lewis, A. E. M. The typical Turing degree. *Proc. Lond. Math. Soc. 109*, 1 (2014), 1–39.

[3] Barmpalias, G., Downey, R., and McInerney, M. Integer valued betting strategies and Turing degrees. *Journal of Computer and System Sciences 81* (2015), 1387–1412.

[4] Barmpalias, G., Downey, R. G., and Ng, K.-M. Jump inversions inside effectively closed sets and applications to randomness. *J. Symbolic Logic 76*, 2 (2011), 491–518.

[5] Bienvenu, L., Stephan, F., and Teutsch, J. How powerful are integer-valued martingales? *Theory of Computing Systems 51*, 3 (2012), 330–351.

[6] Cai, M. A 2-minimal non-GL_2 degree. *J. Math. Log. 10*, 1-2 (2010), 1–30.

[7] Cai, M. A hyperimmune minimal degree and an ANR 2-minimal degree. *Notre Dame J. Form. Log. 51*, 4 (2010), 443–455.

[8] Cai, M. *Elements of Classical Recursion Theory: Degree-Theoretic Properties and Combinatorial Properties*. PhD thesis, Cornell University, 2011.

[9] Cai, M. 2-minimality, jump classes and a note on natural definability. *Ann. Pure Appl. Logic 165*, 2 (2014), 724–741.

[10] Cai, M., Greenberg, N., and McInerney, M. DNR and incomparable Turing degrees. *Forum of Mathematics, Sigma 4* (2016), 44 pages.

[11] Cholak, P., Coles, R., Downey, R. G., and Herrmann, E. Automorphisms of the lattice of Π^0_1 classes: perfect thin classes and anc degrees. *Transactions of the American Mathematical Society 353* (2001), 4899–4924.

[12] Chong, C. T., and Downey, R. Degrees bounding minimal degrees below $0'$. *Proc. Cambridge Phil. Soc 105* (1989), 211–222.

[13] Chong, C. T., and Downey, R. G. Minimal degrees recursive in 1-generic degrees. *Ann. Pure Appl. Logic 48* (1990), 215–225.

[14] Chong, C. T., and Jockusch, Jr., C. G. Minimal degrees and 1-generic sets below $0'$. In *Computation and Proof Theory*

(Aachen, 1983), vol. 1104 of *Lecture Notes in Math.* Springer, 1984, pp. 63–77.

[15] Conidis, C. A measure-theoretic proof of Turing incomparability. *Ann. Pure Appl. Logic 162*, 1 (2010), 83–88.

[16] Downey, R. G., and Greenberg, N. Turing degrees of reals of positive packing dimension. *Information Processing Letters 108* (2008), 198–203.

[17] Downey, R. G., and Greenberg, N. *A transfinite hierarchy of computably enumerable degrees, unifying classes and natural definability*. 2016. Submitted.

[18] Downey, R. G., and Hirshfeldt, D. *Algorithmic Randomness and Complexity*. Springer, 2010.

[19] Downey, R. G., Jockusch, Jr., C. G., and Stob, M. Array nonrecursive sets and multiple permitting arguments. In *In* Recursion Theory Week, Oberwolfach 1989*, eds. K. Ambos-Spies and G. H. Muller and G. E. Sacks*, vol. 1432 of *Lecture Notes in Mathematics*. Springer-Verlag, 1990, pp. 141–174.

[20] Downey, R. G., Jockusch, Jr., C. G., and Stob, M. Array nonrecursive sets and genericity. In *Computability, Enumerability, Unsolvability: Directions in Recursion Theory*, vol. 224 of *London Mathematical Society Lecture Notes Series*. Cambridge University Press, 1996, pp. 93–104.

[21] Downey, R. G., and Miller, J. S. A basis theorem for Π^0_1 classes of positive measure and jump inversion for random reals. *Proc. Amer. Math. Soc. 134*, 1 (2006), 283–288 (electronic).

[22] Downey, R. G., and Stob, M. Minimal pairs in initial segment of the recursively enumerable degrees. *Israel Journal of Mathematics 100*, 1 (1997), 7–27.

[23] Greenberg, N., and Miller, J. S. Diagonally non-recursive functions and effective Hausdorff dimension. *Bull. Lond. Math. Soc. 43*, 4 (2011), 636–654.

[24] Haught, C. A. The degrees below a 1-generic degree $< \mathbf{0}'$. *J. Symbolic Logic 51*, 3 (1986), 770–777.

[25] Ishmukhametov, S. Weak recursive degrees and a problem of Spector. In *Recursion Theory and Complexity, M. Arslanov and S. Lempp, eds.* de Gruyter, Berlin, 1999, pp. 81–88.

[26] Jockusch, Jr., C. G. Degrees of generic sets. In *Recursion Theory: its Generalisations and Applications*, London Mathematical Society Lecutre Note Series. Cambridge University Press, 1980, pp. 110–139.

[27] Jockusch, Jr., C. G. Degrees of functions with no fixed points. In *Logic, methodology and philosophy of science, VIII (Moscow, 1987)*, vol. 126 of *Stud. Logic Found. Math.* North-Holland, Amsterdam, 1989, pp. 191–201.

[28] JOCKUSCH, JR., C. G., AND POSNER, D. Double jumps of minimal degrees. *J. Symbolic Logic 43*, 4 (1978), 715–724.

[29] KHAN, M., AND MILLER, J. S. Forcing with bushy trees. In Preparation.

[30] KUČERA, A. Measure, Π^0_1-classes and complete extensions of PA. In *Recursion theory week (Oberwolfach, 1984)*, vol. 1141 of *Lecture Notes in Math.* Springer, Berlin, 1985, pp. 245–259.

[31] KUČERA, A., AND SLAMAN, T. A. Turing incomparability in Scott sets. *Proc. Amer. Math. Soc. 135*, 11 (2007), 3723–3731.

[32] KUMABE, M., AND LEWIS, A. E. M. A fixed-point-free minimal degree. *J. Lond. Math. Soc. (2) 80*, 3 (2009), 785–797.

[33] KUMMER, M. Kolmogorov complexity and instance complexity of recursively enumerable sets. *SIAM J. Comput. 25*, 6 (1996), 1123–1143.

[34] KURTZ, S. *Randomness and genericity in the degrees of unsolvability.* PhD thesis, University of Illinois at Urbana-Champaign, 1981.

[35] LERMAN, M. *Degrees of Unsolvability.* Springer- Verlag, 1983.

[36] MARTIN-LÖF, P. The definition of random sequences. *Information and Control 9* (1966), 602–619.

[37] NIES, A. *Computability and Randomness.* Oxford University Press, 2009.

[38] Nies, A., Stephan, F., and Terwijn, S. A. Randomness, relativization and Turing degrees. *J. Symbolic Logic 70*, 2 (2005), 515–535.

[39] Sacks, G. E. Some open questions in recursion theory. In *Recursion Theory Week*, G. H. M. . Heinz-Dieter Ebbinghaus and G. E. Sacks, Eds., vol. 1141. Springer, 1985, pp. 333–342.

[40] Schaeffer, B. Dynamic notions of genericity and array noncomputability. *Ann. Pure Appl. Logic 96* (1998), 37–69.

[41] Schnorr., C. P. *Zufälligkeit und Wahrscheinlichkeit. Eine algorithmische Begründung der Wahrscheinlichkeitstheorie.* Volume 218 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin–New York, 1971.

[42] Shore, R. A. The Turing Degrees: An Introduction. In *Forcing, Iterated Ultrapowers, and Turing Degrees*, C. T. Chong, Q. Feng, T. A. Slaman, and W. H. Woodin, Eds., vol. 29 of *Lecture Notes Series, Institute for Mathematical Sciences, National University of Singapore*. World Scientific, 2015.

[43] Soare, R. I. *Recursively Enumerable Sets and Degrees*. Perspectives in Mathematical Logic. Springer-Verlag, 1987.

[44] Terwijn, S., and Zambella, D. Algorithmic randomness and lowness. 1199–1205.

[45] von Mises, R. Grundlagen der wahrscheinlichkeitsrechnung. *Mathematische Zeitschrift 5* (1919), 52–99.